UNIVERSIDAD NACIONAL DE PIURA

Facultad De Ciencias

Escuela Profesional De Ingeniería Electrónica Y Telecomunicaciones

TESIS

"SISTEMA DE PREDICCIÓN DE POSICION Y SEGUIMIENTO DE UNA ESFERA EN TIEMPO REAL UTILIZANDO VISIÓN ARTIFICIAL

PRESENTADA POR:

LUIS MIGUEL RAMOS MOROCHO

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES

LINEA DE INVESTIGACION: Informática, Electrónica y Telecomunicaciones

Sub – Línea de Investigación: Sistemas Digitales

PIURA – PERÚ

2018

Los miembros del Jurado designados para evaluar la tesis presentada por el Bachiller : RAMOS MOROCHO LUIS MIGUEL , titulada:

"SISTEMA DE PREDICCIÓN DE POSICIÓN Y SEGUIMIENTO DE UNA ESFERA EN TIEMPO REAL UTLIZANDO VISIÓN ARTIFICIAL"

Bach. LUIS MIGUEL RAMOS MOROCHO
Ejecutor de Tesis

Mg. JUAN MANUEL JACINTO SANDOVAL
Asesor

Los miembros del Jurado designados para evaluar la tesis presentada por el Bachiller : RAMOS MOROCHO LUIS MIGUEL , titulada:

"SISTEMA DE PREDICCIÓN DE POSICIÓN Y SEGUIMIENTO DE UNA ESFERA EN TIEMPO REAL UTLIZANDO VISIÓN ARTIFICIAL"

Ing. EDUARDO OMAR ÁVILA REGALADO
Presidente

Dr. CARLOS ENRIQUE ARELLANO RAMÍREZ
Secretario

FRANKLIN BAARRA ZAPATA
Vocal

UNIVERSIDAD NACIONAL DE PIURA
FACULTAD DE CIENCIAS

ACTA DE SUSTENTACIÓN 043-2018-D-FC-UNP

Los Miembros del Jurado Calificador que suscriben, reunidos para evaluar la Tesis denominada **"SISTEMA DE PREDICIÓN DE POSICIÓN Y SEGUIMIENTO DE UNA ESFERA EN TIEMPO REAL UTILIZANDO VISIÓN ARTIFICIAL"** presentada por el señor Bachiller **LUIS MIGUEL RAMOS MOROCHO,** con el asesoramiento del **M.Sc. Juan Manuel Jacinto Sandoval;** oídas las observaciones y respuestas a las preguntas formuladas, y de conformidad al Reglamento de Tesis para obtener el Título Profesional en la Facultad de Ciencias, lo declaran:

APROBADO (X) **DESAPROBADO ( )**

Con la mención de:

MUY BUENO

(X) En consecuencia, queda en condición de ser ratificado por el Consejo de Facultad de Ciencias de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES.**

(X) En consecuencia, queda en condición de ser ratificado por el Consejo Universitario de la Universidad Nacional de Piura, y recibir el **TITULO PROFESIONAL DE INGENIERO ELECTRÓNICO Y TELECOMUNICACIONES**; después que el sustentante incorpore la sugerencia del Jurado Calificador.

Piura 20 de julio 2018.

Ing°EDUARDO OMAR ÁVILA REGALADO
PRESIDENTE DE JURADO DE TESIS

DR. CARLOS ENRIQUE ARELLANO RAMÍREZ
SECRETARIO DE JURADO DE TESIS

Ing. FRANKLIN BARRA ZAPATA
VOCAL DE JURADO DE TESIS

UNIVERSIDAD NACIONAL DE PIURA
DECANATO
Facultad de Ciencias

DECLARACIÓN JURADA DE AUTENTICIDAD DE LA TESIS

Yo: LUIS MIGUEL RAMOS MORACHO

identificado con CU/DNI Nº 48419988, Bachiller de Escuela Profesional de Ingeniería Electrónica y Telecomunicaciones, de la Facultad de Ciencias y domiciliado en calle /Jiron/Av. AA.HH VICTOR RAUL MZ F LOTE 19 del Distrito PIURA Provincia PIURA Departamento PIURA Celular: 983923976 Email: luis-94-13@hotmail.com

DECLARO BAJO JURAMENTO: que la tesis que presento es auténtica e inédita, no siendo copia parcial ni total de una tesis desarrollada, y/o realizada en el Perú o en el Extranjero, en caso contrario de resultar falsa la información que proporciono, me sujeto a los alcances de lo establecido en el Art. Nº 411, del código Penal concordante con el Art. 32º de la Ley Nº 27444, y Ley del Procedimiento Administrativo General y las Normas Legales de Protección a los Derechos de Autor.

En fe de lo cual firmo la presente.

Piura, _____

.....

DNI Nº 48419988

Artículo 411.- El que, en un procedimiento administrativo, hace una falsa declaración en relación a hechos o circunstancias que le corresponde probar, violando la presunción de veracidad establecida por ley, será reprimido con pena privativa de libertad no menor de uno ni mayor de cuatro años.

Art. 4. Inciso 4.12 del Reglamento del Registro Nacional de Trabajos de Investigación para optar grados académicos y títulos profesionales –RENATI Resolución de Consejo Directivo Nº 033-2016-SUNEDU/CD

DEDICATORIA

A Dios.

Por darme la oportunidad de vivir y por estar conmigo en cada paso que doy, haberme dado salud para lograr mis objetivos, además de su infinita bondad y amor.

A mi madre Elena.

Por darme la vida, quererme mucho, creer en mí, por sus consejos, sus valores, por la motivación constante que me ha permitido ser una persona de bien, pero más que nada, por su amor. Te amo.

A mi padre Hugo.

Por los ejemplos de perseverancia y constancia que lo caracterizan y que me han infundado siempre, por el valor mostrado para salir adelante y por su amor. Apoyo y animo que me brinda día con día para alcanzar nuevas metas, tanto profesionales como personales.

A mi novia Vanessa.

Tu ayuda ha sido fundamental, has estado conmigo incluso en los momentos más difíciles. Llegar hasta aquí no fue fácil, pero estuviste motivándome y ayudándome hasta donde tus alcances lo permitan, incluso más que eso. Te lo agradezco muchísimo amor.

A mi hermano Rafael.

Que desde el cielo debe estar cuidando y guiando mi camino.

A mi familia

A mi cuñada Rebeca y mis sobrinas Ximena, Rafaela, Valeska, nunca imagine el significado tan grande que tendrían para mí. Las quiero

AGRADECIMIENTO

Este trabajo de tesis primeramente me gustaría agradecerte a ti Dios por bendecirme para llegar hasta donde he llegado, porque hiciste realidad este objetivo.

A mi asesor de tesis, Ing. Juan Manuel Jacinto Sandoval por su esfuerzo y dedicación quien con sus conocimientos, su experiencia, su paciencia y su motivación ha logrado en mí que pueda terminar mis estudios con éxito.

También me gustaría agradecer a mis profesores durante toda mi carrera profesional porque todos has aportado con un granito de arena a mi formación.

Son muchas personas que han formado parte de mi vida profesional a las que me encantaría agradecerles su amistad, consejos, apoyo, ánimo y compañía en los momentos más difíciles de mi vida. Algunas están aquí conmigo y otras en mi recuerdo y en mi corazón, sin importar en donde estén quiero darles las gracias por formar parte de mí, por todo lo que me han brindado y por todas sus bendiciones.

A todos ustedes, mi mayor reconocimiento y gratitud.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

La visión artificial es una disciplina científica que incluye métodos para adquirir, procesar y analizar imágenes del mundo real con el fin de producir información que pueda ser tratada por una máquina. Se estudia el problema de seguimiento y estimación de las trayectorias realizadas por una esfera en movimiento en un escenario. La entrada al sistema la constituyen imágenes capturadas de la escena de interés.

Se analizan las etapas del sistema y los algoritmos que deben aplicarse en cada una de ellas para un caso concreto, donde el movimiento de la esfera es rectilíneo uniforme.

El reconocimiento de las trayectorias realizadas por la esfera se efectúa en función de las posiciones ocupadas por ésta a lo largo de la secuencia de frames buscando minimizar una función que determina la desviación de una trayectoria. La trayectoria de cada esfera se actualiza con cada nuevo frame que ingresa al sistema.

La estimación del recorrido futuro de un objeto se realiza a partir de su recorrido previo modelando el movimiento en cada uno de los ejes de coordenadas mediante una función lineal.

Para la adquisición de las imágenes se utilizará una cámara webcam con conexión usb, el procesamiento se realizará empleando el lenguaje de programación Python y las librerías de visión por computadora OpenCv.

Vision Artificial, Python, estimación de trayectoria, OpenCv.

ABSTRACT

Artificial vision is a scientific discipline that includes methods to acquire, process and analyze real-world images in order to produce information that can be treated by a machine. The problem of tracking and estimating the trajectories made by a sphere in motion in a scenario is studied. The entrance to the system is captured images of the scene of interest.

The stages of the system and the algorithms that must be applied in each one of them are analyzed for a concrete case, where the movement of the sphere is uniform rectilinear.

The recognition of the trajectories carried out by the sphere is carried out in function of the positions occupied by it throughout the sequence of frames seeking to minimize a function that determines the deviation of a trajectory. The trajectory of each sphere is updated with each new frame that enters the system.

The estimation of the future path of an object is made from its previous travel modeling the movement in each of the axes of coordinates by a linear function.

For the acquisition of the images a webcam webcam with USB connection will be used, the processing will be done using the Python programming language and the OpenCv computer vision libraries.

Artificial Vision, Python, trajectory estimation, OpenCv.

INTRODUCCIÓN

Existe una amplia gama de aplicaciones en donde es necesario resolver el problema de predicción de posición y seguimiento realizadas por un conjunto de entidades u objetos en movimiento. Entre ellas podemos mencionar el guiado asistido de vehículos, el seguimiento de misiles mediante el uso de radares, el análisis del movimiento de espermatozoides en una muestra de semen, las competencias de fútbol robótico, etc.

Por seguimiento de trayectorias nos referimos a la acción de identificar el recorrido realizado por cada una de las entidades de interés dentro del escenario analizado. La predicción de posición consiste en utilizar la información correspondiente a los recorridos identificados junto con un modelo del movimiento de los objetos para predecir sus recorridos futuros. El último paso consiste en utilizar la información de las trayectorias armadas hasta el momento para predecir como continuará cada una de ellas. El error cometido en la estimación de la trayectoria de un objeto depende de diversos factores. En primer lugar, está influenciado por la información que se disponga acerca de la trayectoria previa. Es claro que cuanto más exacta sea dicha información, los resultados obtenidos a partir de ella serán más precisos. Por otro lado, cuanto mayor conocimiento se disponga acerca de las características del movimiento de un objeto, se podrá realizar una estimación más exacta de su movimiento. Si un objeto se mueve realizando una trayectoria circular, y se predice el recorrido futuro suponiendo que la trayectoria es lineal, la estimación resultará en un fracaso. Por lo tanto, es importante el conocimiento que se disponga acerca de las características del movimiento para poder modelarlo con la mayor exactitud posible. Es importante destacar que las técnicas que se aplican en cada una de las etapas mencionadas varían notablemente de acuerdo con las características de cada problema concreto.

En el caso general, los algoritmos para resolver el problema de predicción de posición implican una alta carga de procesamiento, además de que se deben considerar restricciones de tiempo real, y la velocidad de la arquitectura de cómputo puede no ser suficiente para realizar un muestreo adecuado para la estimación. Por estos motivos, surge naturalmente la posibilidad de utilizar un sistema paralelo para la resolución del problema. Este hecho se acentúa si se considera el caso de contar con más de una cámara.

I. ASPECTOS DE LA PROBLEMATICA

1.1. DESCRIPCIÓN DE LA REALIDAD PROBLEMÁTICA.

El presente proyecto se basa principalmente en el desarrollo de un sistema automatizado, mediante el uso de visión artificial. El seguimiento de objetos es uno de los problemas críticos dentro de los procesos de visión artificial donde la exactitud y el tiempo de respuesta son de gran importancia dentro de la respuesta esperada, por ejemplo, en los sistemas de vigilancia, en la robótica y en la industria cuando se necesita hacer seguimiento a un proceso. Los sistemas de visión por computador se basaron inicialmente en imágenes estáticas, sin embargo, el mundo es dinámico, por tanto, el esfuerzo invertido en mejorar la habilidad de seguir objetos en movimiento queda justificado. Actualmente existen diversas aplicaciones, siendo la robótica y la medicina dos de las más importantes.

En la mayoría de las tareas de localización de objetos mediante visión artificial los objetos son estáticos. Existen diversos factores que determinan la dificultad en la solución del problema de seguimiento de objetos y un aspecto fundamental al desarrollar estructuras de representación de objetos es su forma. Los objetos pueden estar constituidos por puntos, líneas, aristas, curvas o poseer una morfología libre, incrementando el grado de dificultad al identificar el objeto.

¿Será posible implementar un sistema de predicción de posición y seguimiento de una esfera en tiempo real utilizando visión artificial?

1.2. JUSTIFICACIÓN E IMPORTANCIA DE LA INVESTIGACIÓN

El seguimiento de objetos por medio de visión artificial proporciona al sistema la información básica del ambiente por medio de imágenes. El proceso de captura del sistema visual a través de una cámara recibe a su entrada una secuencia de imágenes dinámica tomada del mundo real cada una de ellas en un instante de tiempo diferente. Se busca con este trabajo abordar el problema de seguimiento de objetos cuando están en movimiento utilizando una secuencia de imágenes, haciendo uso de visión por computadora y algoritmos de procesamiento de imágenes y video. Al implementar sistemas de seguimiento de objetos a través de

captura de imágenes con un buen procesador y algoritmos suficientes se tiene una alta velocidad de procesamiento.

1.3. OBJETIVOS

OBJETIVO GENERAL

Diseñar un sistema de predicción de posición y seguimiento de una esfera en tiempo real utilizando visión artificial.

OBJETIVOS ESPECÍFICOS

- ✓ Adquirir la secuencia de imágenes utilizando una cámara webcam
- ✓ Segmentar la esfera del fondo de la imagen, utilizando el modelo HSV.
- ✓ Implementar el algoritmo de predicción de posición de objetos.
- ✓ Validar el algoritmo desarrollado con visualización en pantalla de las imágenes procesadas del seguimiento del objeto.
- ✓ Desarrollar el sistema utilizando el lenguaje de programación Python.

1.4. DELIMITACIÓN DE LA INVESTIGACIÓN

El proyecto será desarrollado, en los ambientes de la Universidad Nacional de Piura.

Se utilizará una correcta iluminación, no se tomará en cuenta ambientes oscuros. La esfera será de color roja sobre un fondo uniforme de un color diferente al de la esfera, de preferencia color blanco.

La cámara se colocará de forma fija o estática en un plano paralelo y a una determinada altura de la escena.

II. MARCO TEORICO

2.1. ANTECEDENTES DE LA INVESTIGACIÓN

En la tesis: "SEGUIMIENTO EN TIEMPO REAL DE OBJETOS SOBRE SECUENCIA DE IMÁGENES EMPLEANDO DISPOSITIVOS DE LÓGICA PROGRAMABLE" presentada por MARLON STEEPHEN NARVÁEZ RODRIGUEZ y MIGUEL ANDRÉS SARRIA FERNÁNDEZ, de la UNIVERSIDAD TECNOLÓGICA DE PEREIRA, FACULTAD DE INGENIERÍAS: ELÉCTRICA, ELECTRÓNICA, FÍSICA Y CIENCIAS DE LA COMPUTACIÓN PROGRAMA DE INGENIERÍA ELECTRÓNICA, PEREIRA, COLOMBIA, han desarrollado un sistema de grado donde se plantea una metodología para determinar, en tiempo real, la ubicación y orientación de un objeto en movimiento, sobre una superficie plana, durante un intervalo de tiempo. Se utilizaría para ello una cámara y una tarjeta de video para la adquisición de las imágenes, el procesamiento se realizaría empleando dispositivos de lógica programable con visualización en pantalla. Los principales problemas al realizar este tipo de aplicaciones es el alto costo computacional y económico de los elementos que se utilizan para tal, como también de la velocidad de respuesta del sistema a implementar. Se propone una solución de compromiso entre la precisión en la determinación de la información obtenida, el costo computacional y la velocidad de la técnica. Con la técnica a desarrollar se pretende en primer lugar realizar la adquisición de la imagen, el preprocesamiento y procesamiento de la imagen, la identificación del objeto para luego abordar el problema de seguimiento de este y finalmente su visualización. En la parte de identificación se utiliza segmentación por color y una técnica para el seguimiento del objeto.

En la tesis: "TÉCNICAS DE PROCESADO DE IMAGEN PARA EL SEGUIMIENTO DE OBJETOS DESDE VEHÍCULOS AÉREOS NO TRIPULADOS" presentada por Natalia Ibáñez Sáez, de la UNIVERSIDAD DE VALLADOLID, del Departamento de Teoría de la Señal y Comunicaciones e Ingeniería Telemática para obtener el GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN, han realizado una herramienta de procesamiento de imagen capaz de realizar la detección y seguimiento de objetos en movimiento grabados desde vehículos aéreos no tripulados. El problema de realizar video vigilancia desde cámaras móviles, como sucede en el

caso de vehículos aéreos no tripulados, supone retos adicionales a los existentes en situaciones en las que la cámara está fija. Por ello, varios de los métodos utilizados en el caso de cámara fija no son aplicables cuando la cámara está en movimiento. En el estado del arte se han analizado varios de los métodos utilizados en el problema de video vigilancia con cámara móvil y se ha seleccionado el método basado en alineamiento de la imagen mediante puntos característicos basado en flujo óptico por ser un método que proporciona buenos resultados y cuyo tiempo de ejecución es relativamente bajo.

El método seleccionado para realizar la herramienta deseada tiene como parte fundamental la comparación de dos flujos ópticos: un flujo óptico real, que describe el movimiento que se produce en los puntos seleccionados y un flujo óptico artificial, que describe el movimiento global de la imagen y que corresponde al movimiento del fondo de la imagen. Mediante la comparación de ambos flujos es posible compensar el movimiento producido por la cámara móvil y que, de este modo, todo el movimiento que recoja la cámara corresponda a los objetos en movimiento. Así mismo, es fundamental realizar un posterior seguimiento de los objetos móviles que permite realizar un filtrado espacial y temporal de los mismos, eliminando posibles falsas detecciones de objetos móviles producidos por errores en la fase de detección. Este seguimiento proporciona también información sobre la trayectoria de los objetos y se realiza mediante el filtro de Kalman.

Por último, la herramienta realizada se ha probado con dos bancos de vídeos diferentes. El primero de ellos corresponde a una serie de vídeos grabados con una cámara de fotos en distintas situaciones de iluminación, movimiento de la cámara y en presencia de diferente número de objetos. Estos vídeos han sido utilizados para fijar los parámetros de funcionamiento de la herramienta y realizar un análisis cuantitativo que refleje la influencia de las características del vídeo en el funcionamiento de la herramienta, así como el tiempo que tarda la herramienta en inicializarse. El segundo banco de vídeos consiste en vídeos grabados por un vehículo aéreo no tripulado controlado de forma teledirigida. Estos vídeos han sido utilizados para evaluar cualitativamente el buen funcionamiento de la herramienta a través de la comprobación visual de los vídeos procesados.

2.2. BASES TEÓRICAS

2.2.1. REGRESIÓN LINEAL

El análisis de regresión tiene principalmente tres objetivos, predicción, modelamiento y caracterización, los cuales, en diversas ocasiones, pueden estar completamente ligados.

La idea detrás de un análisis de regresión es encontrar una relación funcional entre un conjunto de datos dado, aproximando su comportamiento a una función predefinida con parámetros variables y dentro de un intervalo definido. Dependiendo de esta función, aparecen diferentes métodos de regresión los cuales varían principalmente en complejidad o error de aproximación siendo estas dos variables inversamente proporcionales. Cuando se obtiene la función que relaciona los datos, se puede aprender mucho más acerca del fenómeno que los produjo e inclusive se pueden llegar a dar predicciones aproximadas de lo que pueda pasar en tiempos futuros.

Entre los muchos métodos de regresión existentes, se escoge regresión lineal por su fácil implementación y menor costo computacional.

Suponiendo la existencia de un conjunto de datos X y Y, se asume una relación funcional entre ellos de la forma:

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

donde β_0 y β_1 son los parámetros del modelo y ε es el incremento por el cual cada valor de Y puede desviarse de la línea de regresión.

Se asume que este modelo se mantiene mientras se desarrolla el método, pero son necesarias etapas siguientes para saber si en verdad lo hace o no.

β_0 y β_1 son desconocidos, pero se mantienen fijos, lo cual permite hallar valores estimados bo y b1 de β_0 y β_1 a partir de los datos iniciales:

$$\hat{Y} = b_0 + b_1 X$$

Para hallar los valores estimados, se utiliza el método de mínimos cuadrados el cual consiste en hallar los valores que hacen mínimo el error cuadrático.

Adicionando al modelo lineal todo el conjunto de valores, se tiene que la suma de cuadrados de las desviaciones es:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon$$

$$S = \sum_{i=1}^{n} \varepsilon_i^2 = \sum_{i=1}^{n} (Y_i - \beta_0 - \beta_1 X_i)^2$$

Se escogen los valores estimados bo y b1 para los cuales, cuando se reemplacen por β_0 y β_1 produce el menor valor posible de S.

Diferenciando la suma de cuadrados por β_0 y β_1 e igualando el resultado a cero:

$$\frac{\partial S}{\partial \beta_0} = -2\sum_{i=1}^{n} (Y_i - \beta_0 - \beta_1 X_i)$$

$$\sum_{i=1}^{n} (Y_i - b_0 - b_1 X_i) = 0$$

$$\frac{\partial S}{\partial \beta_1} = -2\sum_{i=1}^{n} (Y_i - \beta_0 - \beta_1 X_i)$$

$$\sum_{i=1}^{n} X_i (Y_i - b_0 - b_1 X_i) = 0$$

Despejando las ultimas ecuaciones, se tienen las ecuaciones normales.

$$b_0 n + b_1 \sum_{i=1}^{n} X_i = \sum_{i=1}^{n} Y_i$$

$$b_0 \sum_{i=1}^{n} X_i + b_1 \sum_{i=1}^{n} X_i^2 = \sum_{i=1}^{n} X_i Y_i$$

Donde:

$$b_1 = \frac{\sum X_i Y_i - \left[\left(\sum X_i\right)\left(\sum Y_i\right)\right]/n}{\sum X_i^{\,2} - \left(\sum X_i\right)^2/n} = \frac{\sum \left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)}{\sum \left(X_i - \overline{X}\right)^2}$$

Se define:

$$S_{XY} = \sum \left(X_i - \overline{X}\right)\left(Y_i - \overline{Y}\right)$$

$$S_{XX} = \sum \left(X_i - \overline{X}\right)^2$$

$$S_{XX} = \sum \left(Y_i - \overline{Y}\right)^2$$

Con lo cual se obtiene:

$$b_1 = S_{XY}/S_{XX} \qquad b_0 = \overline{Y} - b_1\overline{X}$$

2.2.2. MINIMOS CUADRADOS

Es un procedimiento de análisis numérico en la que, dados un conjunto de datos (pares ordenados y familia de funciones), se intenta determinar la función continua que mejor se aproxime a los datos (línea de regresión o la línea de mejor ajuste), proporcionando una demostración visual de la relación entre los puntos de

estos. En su forma más simple, busca minimizar la suma de cuadrados de las diferencias ordenadas (llamadas residuos) entre los puntos generados por la función y los correspondientes datos.

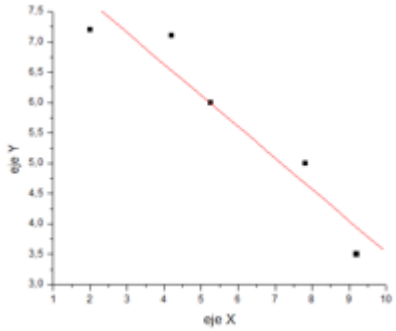

FIGURA 1 CONJUNTO DE PARES ORDENADOS

Este método se utiliza comúnmente para analizar una serie de datos que se obtengan de algún estudio, con el fin de expresar su comportamiento de manera lineal y así minimizar los errores de la data tomada.

La creación del método de mínimos cuadrados generalmente se le acredita al matemático alemán Carl Friedrich Gauss, quien lo planteó en 1794 pero no lo publicó sino hasta 1809. El matemático francés Andrien-Marie Legendre fue el primero en publicarlo en 1805, este lo desarrolló de forma independiente.

2.2.2.1. DEFINICIÓN: MÉTODO DE MÍNIMOS CUADRADOS

Su expresión general se basa en la ecuación de una recta y = mx + b. Donde m es la pendiente y b el punto de corte, y vienen expresadas de la siguiente manera:

$$m = \frac{n \cdot \Sigma(x \cdot y) - \Sigma x \cdot \Sigma y}{n \cdot \Sigma x^2 - |\Sigma x|^2}$$

$$b = \frac{\Sigma y \cdot \Sigma x^2 - \Sigma x \cdot \Sigma(x \cdot y)}{n \cdot \Sigma x^2 - |\Sigma x|^2}$$

Σ es el símbolo sumatorio de todos los términos, mientas (x, y) son los datos en estudio y n la cantidad de datos que existen.

El método de mínimos cuadrados calcula a partir de los N pares de datos experimentales (x, y), los valores m y b que mejor ajustan los datos a una recta. Se entiende por el mejor ajuste aquella recta que hace mínimas las distancias d de los puntos medidos a la recta.

Teniendo una serie de datos (x, y), mostrados en un gráfico o gráfica, si al conectar punto a punto no se describe una recta, debemos aplicar el método de mínimos cuadrados, basándonos en su expresión general:

$$y = \left(\frac{n \cdot \Sigma(x \cdot y) - \Sigma x \cdot \Sigma y}{n \cdot \Sigma x^2 - |\Sigma x|^2}\right) x + \left(\frac{\Sigma y \cdot \Sigma x^2 - \Sigma x \cdot \Sigma(x \cdot y)}{n \cdot \Sigma x^2 - |\Sigma x|^2}\right)$$

Cuando se haga uso del método de mínimos cuadrados se debe buscar una línea de mejor ajuste que explique la posible relación entre una variable independiente y una variable dependiente. En el análisis de regresión, las variables dependientes se designan en el eje y vertical y las variables independientes se designan en el eje x horizontal. Estas designaciones formarán la ecuación para la línea de mejor ajuste, que se determina a partir del método de mínimos cuadrados.

2.2.3. VISION ARTIFICIAL

También conocida como visión por computador (del inglés computer vision) o visión técnica, es un subcampo de la inteligencia artificial y es el campo de acción más ambicioso del procesamiento digital de imágenes. Básicamente el objetivo es automatizar funciones de inspección visual, tradicionalmente utilizadas por el hombre como, por ejemplo.

Los objetivos típicos de la visión artificial incluyen:

La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo, caras humanas).

La evaluación de los resultados (ej.: segmentación, registro).

Registro de diferentes imágenes de una misma escena u objeto, hacer concordar un mismo objeto en diversas imágenes.

Seguimiento de un objeto en una secuencia de imágenes.

Mapeo de una escena para generar un modelo tridimensional de la escena; tal modelo podría ser usado por un robot para navegar por la escena.

Estimación de las posturas tridimensionales de humanos.

Búsqueda de imágenes digitales por su contenido.

2.2.4. PROCESAMIENTO DE IMAGENES[1]

Imagen: Es la proyección en perspectiva en el plano bidimensional de una escena tridimensional en un determinado instante de tiempo t_0.

Fotograma: Es una matriz bidimensional de valores de intensidad lumínica obtenidos para un tiempo t0 constante. Pudiera decirse en cierta forma que es una imagen discretizada.

Píxel (Picture Element): Es cada de una de las posiciones en que es discretizada una imagen, o lo que es lo mismo, cada una de las posiciones de un cuadro.

Imagen binaria: Son aquellas imágenes cuyos píxeles solo tienen dos valores: cero y uno.

[1] Rafael C. Gonzales, Richard E. Woods, "Tratamiento Digital de Imágenes"

2.2.5. LA IMAGEN DIGITAL[2]

Son el principal ingrediente de lo que se conoce como Visión Artificial y representan mediante algún tipo de codificación, normalmente en una matriz de números de dos dimensiones, una escena del entorno.

Existen dos tipos de imágenes utilizadas frecuentemente en Visión Artificial: imágenes de intensidad e imágenes de alcance (también llamadas imágenes de profundidad o perfiles de superficie). Las imágenes de intensidad miden la cantidad de luz que incide en un dispositivo fotosensible, mientras que las imágenes de alcance estiman directamente la estructura en tres dimensiones (3D) de la escena ya que su fundamento radica en el uso de sensores de alcance ópticos y algún fenómeno físico para adquirir la imagen. Un ejemplo típico de una imagen de intensidad es una fotografía, mientras que de una imagen de alcance es, por ejemplo, la imagen que obtiene el oftalmólogo sobre el grado de rugosidad de la córnea de un paciente o las imágenes de un radar.

Aunque la filosofía de los diferentes tipos de imágenes es diferente, en cualquier caso, tras su captura tendremos una matriz de valores en dos dimensiones (2D), es decir, una imagen digital.

2.2.6. DISPOSITIVOS DE CAPTURA DE IMÁGENES

Para la adquisición de imágenes digitales se requieren dos elementos básicos. El primero es un dispositivo físico que es sensible a una determinada banda del espectro de energía electromagnético (tal como rayos X, ultravioleta, visible, infrarrojo, etc.) y que produce una señal eléctrica de salida proporcional al nivel de energía incidente en cualquier instante de tiempo. El segundo, denominado digitalizador, es un dispositivo que cumple la función de convertir la señal eléctrica continua de salida del dispositivo físico en un conjunto discreto de localizaciones del plano de la imagen y, después, en la cuantización de dicha muestra. Esto implica, en primer lugar, determinar el valor de la imagen continua en cada una de las diferentes localizaciones discretas de la imagen (cada valor

[2] Gonzalo Pajares Martinsanz, Jesús Manuel de la Cruz García, José Manuel molina pascual, Juan Cuadrado Pardo, Alejandro López Correa, “Imágenes digitales - Procesamiento practico con java”

localizado de forma discreta se denomina muestra de la imagen) y, luego, asignar a cada muestra una etiqueta entera discreta, que es representativa del rango en el que varía la muestra.

Una vez capturada la señal continua y cuantificada espacialmente y en amplitud, se obtiene una imagen digital, que es como se representa en el computador, es decir, tendremos la matriz (2D) de números como ya hemos mencionado anteriormente. Éstos son los valores que se manipulan para extraer información de las imágenes mediante programas (software).

En la siguiente figura se ilustra un ejemplo de la cuantización espacial y en amplitud:

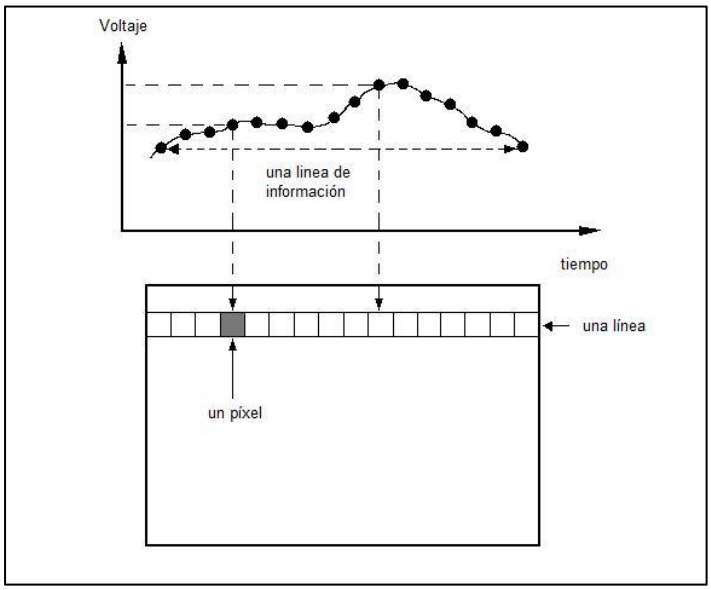

FIGURA 2 DIGITALIZACIÓN DE UNA SEÑAL ANALÓGICA

Supongamos que tenemos una señal analógica, que bien podría ser una línea de video analógica. Esta señal (línea) analógica de video se convierte a una imagen digital muestreando la señal analógica a intervalos determinados. El procedimiento consiste en medir el voltaje de la señal a intervalos de tiempo fijos. El valor del voltaje en cada instante se convierte a un número que es almacenado y se corresponde con la intensidad de la imagen en ese punto. La intensidad en cada punto depende tanto de las propiedades intrínsecas del objeto que se está

viendo como de las condiciones de luz de la escena. Repitiendo este procedimiento para todas las líneas de video que constituyen una imagen, se pueden grabar los resultados obtenidos en el computador, de suerte que habremos conseguido una imagen digital que, en definitiva, es una matriz de números.

La imagen puede accederse como una matriz bidimensional (2D) de datos, donde cada punto o dato se denomina píxel.

Además de las cámaras de televisión que generan una imagen de video, uno de los sensores más usados para la visión artificial son loa dispositivos de acoplamiento de carga (Charge Coupled Devices - CCD). Entre los dispositivos CCD, que generalmente también producen una señal continua de video, cabe distinguir dos categorías: sensores de exploración de línea y sensores de exploración de área. Estos sensores CCD se basan en unos elementos semiconductores llamados photosites. Los fotones procedentes de la escena excitan el elemento semiconductor, de forma que el grado de excitación es proporcional a la cantidad de carga acumulada en el photosite y, por lo tanto, a la intensidad luminosa en ese punto.

Estos photosites se pueden representar en forma de matriz como se muestra en la siguiente figura:

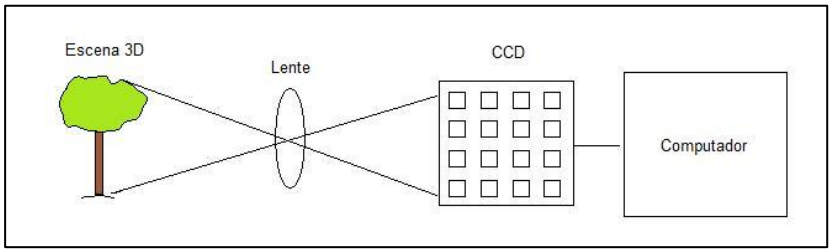

FIGURA 3 CAPTURA DE UNA IMAGEN 3D POR UN DISPOSITIVO CCD

Supongamos una matriz de photosites situados detrás de una lente y sobre los que se proyecta una imagen procedente de la escena 3D. La señal de estos sensores se procesa en el propio sensor (cámara) o en otro dispositivo (tarjeta de procesamiento de imágenes digitales) y los valores digitales se envían al computador.

Para clarificar un poco más estos conceptos, analicemos el ejemplo sencillo mostrado en la figura siguiente:

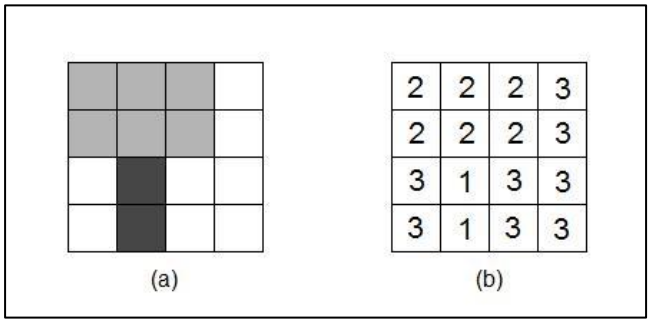

FIGURA 4 FIGURA DEL ÁRBOL CAPTURADA POR UNA CÁMARA

CON 4X4 SENSORES DE INTENSIDAD

Supongamos una escena 3D cuya representación digital en forma de matriz, tal como se almacena en el computador, resulta ser la que se muestra en la figura. En este caso se utiliza una matriz de 16 elementos (4x4) para representar el árbol; lógicamente, cuantos más sensores dispongamos en la cámara, más precisión podemos tener en la reproducción de la imagen. Esto es lo que se conoce como resolución espacial de la imagen.

2.2.7. IMÁGENES BLANCO/NEGRO Y COLOR

En definitiva, e independientemente del tipo de sensor utilizado, la imagen que ha de ser tratada por el computador se presenta digitalizada espacialmente en forma de matriz con una resolución de M x N elementos.

Si la imagen es en Blanco y Negro (B/N), se almacena un valor por cada píxel. Se suele utilizar un rango de valores para su representación, que generalmente es de 0 a 2n-1. Uno de los valores más utilizados de n es 8; esto significa que el rango de valores para este caso varía de 0 a 255. En este caso, el 0 representa el negro absoluto y el 255, el blanco absoluto. Esto indica que podemos tener una resolución o precisión en los grises posibles de 256. El hecho

de utilizar 256 niveles es porque con 8 bits del computador se pueden codificar 256 valores distintos desde la combinación 00000000, que representa el nivel 0, hasta la combinación 11111111, que representa el nivel 255.

En el caso de las imágenes en color, los elementos de la matriz vienen dados por tres valores, que representan cada uno de los componentes básicos del color en cuestión. Estos componentes son el Rojo (R), Verde (G) y Azul (B), el conocido código RGB. En este caso el conjunto de valores (0,0,0) es el negro absoluto; el (255,255,255), el blanco absoluto; el (255,0,0), el rojo puro; el (0,255,0), el verde puro; el (0,0,255), el azul puro. Como es lógico, la combinación de distintos valores proporciona otros colores, por ejemplo, el (255, 255, 51) es un tono de amarillo o el (204, 153, 102), es un tono marrón. El número de colores posible resulta ser 255.

2.2.8. RESOLUCIÓN ESPACIAL Y EN AMPLITUD

Para comprender mejor el sentido y la diferencia que existe entre resolución espacial y en amplitud, vamos a ilustrar los conceptos en las siguientes figuras:

FIGURA 5 CUATRO REPRESENTACIONES DE LA MISMA IMAGEN

CON VARIACIÓN EN EL NÚMERO DE PÍXELES UTILIZADOS

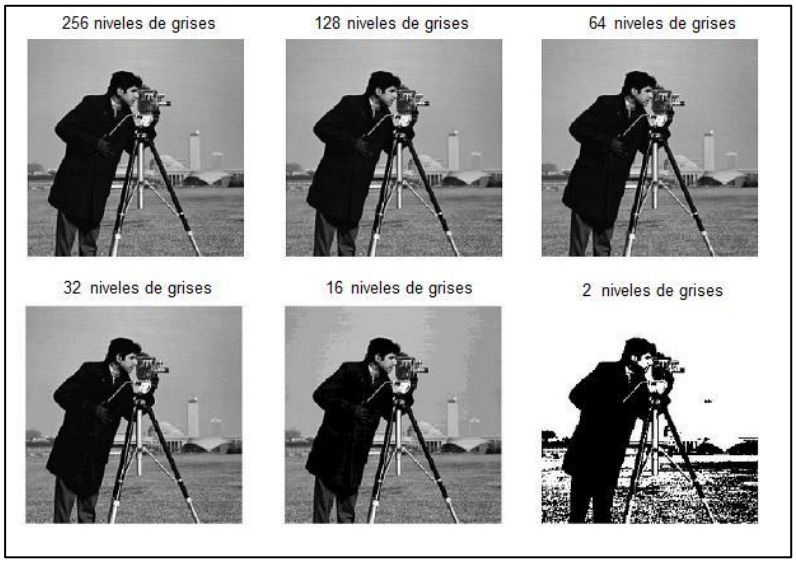

FIGURA 6: SEIS REPRESENTACIONES DE LA MISMA IMAGEN CON VARIACIÓN EN EL NÚMERO DE NIVELES DE GRIS UTILIZADOS

Como se muestra en las figuras anteriores, dependiendo del número de píxeles que tenga el dispositivo y de niveles de grises con que se trabaje en el computador, la imagen poseerá más o menos resolución espacial y en amplitud respectivamente.

2.2.9. REPRESENTACIÓN DE IMÁGENES DIGITALES

Como ya se ha mencionado antes, el termino imagen se refiere a una función de intensidad bidimensional, la cual puede ser representada como f(x,y),f(i,j),I(x,y),I(i,j), etc., donde x e y, o bien, i y j son las coordenadas espaciales y el valor de f o I en cualquier punto (x,y) o (i,j) es proporcional a la intensidad o nivel de gris de la imagen en ese punto.

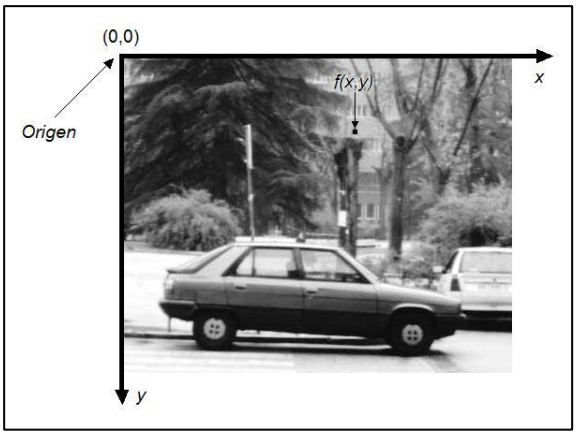

FIGURA 7: CONVENCIÓN DE EJES UTILIZADA PARA LA REPRESENTACIÓN DE IMÁGENES DIGITALES

2.2.10. PROCESAMIENTO Y ANALISIS DE IMÁGENES DIGITALES

Aunque la distinción entre procesamiento y análisis de imágenes digitales no es obvia de forma inmediata, el procesamiento de imágenes puede ser visto como una transformación de una imagen a otra imagen, es decir, a partir de una imagen, se obtiene otra imagen modificada. Por otro lado, el análisis es una transformación de una imagen en algo distinto a una imagen; en consecuencia, el análisis es un determinado tipo de información representando una descripción o una decisión. En la mayoría de los casos, las técnicas de análisis de imágenes digitales son aplicadas a imágenes que han sido procesadas previamente. Desde el punto de vista de un observador humano, el análisis de imágenes es una tarea fácil y rápida, algo que no ocurre en visión artificial. Igualmente, la capacidad de percepción humana permite procesar rápidamente una imagen para detectar en ella características de interés, por ejemplo, bordes o regiones.

2.2.11. PROCESAMIENTO BASICO DE IMÁGENES

El procesamiento de datos en el sistema de visión puede enfocarse desde 2 perspectivas:

Alteración píxel a píxel de los datos en una escala global (individuales).

Operaciones basadas en múltiples puntos (vecindad).

La generación de un nuevo píxel en una imagen será una función bien del valor de cada píxel en su localización individual, o bien de los valores de los píxeles en la vecindad de un píxel dado, como se indica en la figura siguiente:

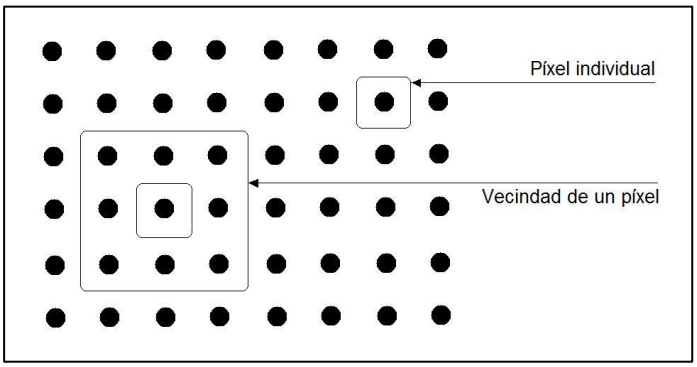

FIGURA 8: FUNCIONES DE PUNTO Y VECINDAD

Existen aún otras operaciones, que no se clasifican ni como individuales ni como vecindad ya que trasforman las imágenes por otros procedimientos; son transformaciones que operan globalmente sobre los valores de intensidad de la imagen cuyo efecto es un realzado de la imagen original, operaciones aritméticas y lógicas, estas últimas basadas en la teoría del algebra de Boole, y operaciones que realizan transformaciones geométricas, sin modificar los valores de intensidad.

2.2.11.1. OPERACIONES INDIVIDUALES

Las operaciones individuales implican la generación de una nueva imagen modificando el valor del pixel en una simple localización basándose en una regla global aplicada a cada localización de la imagen original. El proceso consiste en obtener el valor del píxel de una localización dada en la imagen, modificándolo por una operación lineal o no lineal y colocando el valor del nuevo pixel en la correspondiente localización de la imagen nueva. El proceso se repite para todas y cada de las localizaciones de los pixeles en la imagen original.

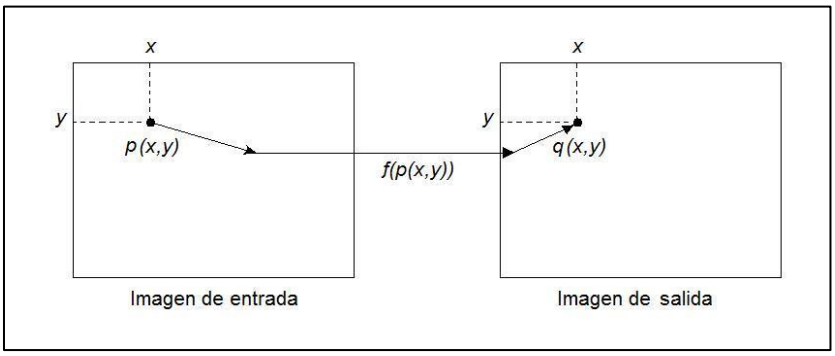

FIGURA 9: OPERACIÓN INDIVIDUAL

Como se aprecia en la figura anterior, el operador individual es una transformación uno a uno. El operador f se aplica a cada píxel en la imagen o sección de la imagen y la salida depende únicamente de la magnitud del correspondiente píxel de entrada; la salida es independiente de los píxeles adyacentes. La función transforma el valor del nivel de gris de cada píxel en la imagen y el nuevo valor se obtiene a través de la ecuación:

$$q(x,y) = f(p(x,y))$$

La función f puede ser un operador lineal o no lineal. El proceso matemático es relativamente simple. La imagen resultante es de la misma dimensión que la original.

2.2.11.2. OPERADOR IDENTIDAD

Este operador crea una imagen de salida que es idéntica a la imagen de entrada. La función de transformación es:

$$q=p$$

El operador identidad deja la imagen de entrada invariante. En la siguiente figura se muestra la función de transformación dada por la ecuación anterior:

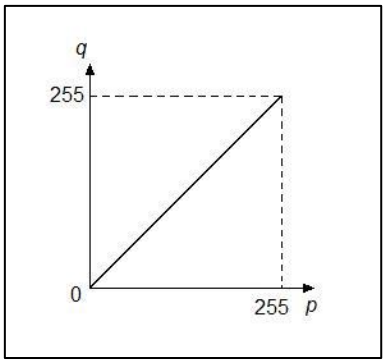

FIGURA 10: REPRESENTACIÓN DEL OPERADOR IDENTIDAD

2.2.11.3. OPERADOR INVERSO O NEGATIVO

Este operador crea una imagen de salida que es inversa de la imagen de entrada. Este operador es útil en diversas aplicaciones tales como imágenes médicas. Para una imagen con valores de gris en el rango de 0 a 255 la función de transformación resulta ser:

$$q = 255 - p$$

En la siguiente figura se muestra la función de transformación dada por la ecuación anterior:

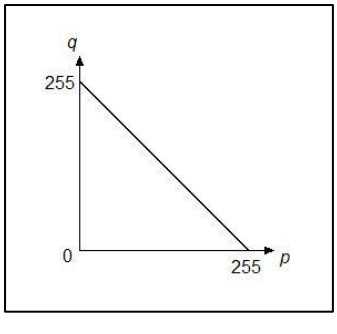

FIGURA 11: REPRESENTACIÓN DEL OPERADOR INVERSO

2.2.11.4. OPERADOR UMBRAL

Esta clase de transformación crea una imagen de salida binaria a partir de una imagen de grises, donde el nivel de transición está dado por el parámetro de entrada p_1. La función de transformación es la siguiente:

$$q = \begin{cases} 0 & para\ p \leq p_1 \\ 255 & para\ p > p_1 \end{cases}$$

En la siguiente figura se muestra la función de transformación dada por la ecuación anterior:

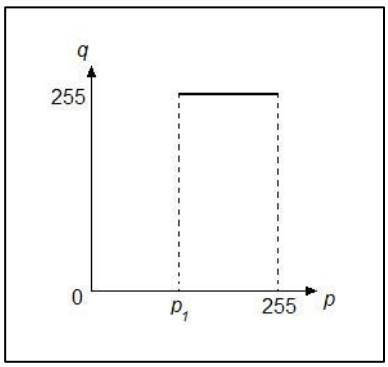

FIGURA 12: REPRESENTACIÓN DEL OPERADOR UMBRAL

2.2.12. TRANSFORMACIÓN DE VECINDAD

En las operaciones de vecindad modifican el nuevo valor del píxel en la imagen de salida depende de una combinación de los valores de los píxeles en la vecindad de la imagen original que está siendo transformada.

Dentro de la categoría de operaciones de vecindad se incluyen las operaciones de filtrado. Las operaciones de filtrado tienen la particularidad de eliminar un determinado rango de frecuencias de las imágenes.

2.2.12.1. Nociones y propiedades de vecindad

Se dice que todo píxel p, de coordenadas (x,y), tiene cuatro píxeles que establecen con él una relación de vecindad horizontal o vertical, que son:

Horizontal: (x-1, y) y (x+1, y) Vertical: (x, y-1) y (x, y+1)

Estos cuatro pixeles definen lo que se conoce como entorno de vecindad-4 y nos referimos a ellos como E_4 (p).

Los cuatro vecinos diagonales de p tienen coordenadas:

(x-1, y-1), (x+1, y-1), (x-1, y+1), (x+1, y+1)

y nos referimos a ellos como E_D (p). Estos píxeles junto con los E_4 (p) se llaman los vecinos-8 de p y se denotan como E_8 (p).

Existen excepciones dadas cuando el píxel (x, y) es un punto del borde de la imagen, en cuyo caso algunos de los vecinos definidos anteriormente no existen.

2.2.12.2. Conectividad

Sea V el conjunto de valores de intensidad de los píxeles que se permiten estén adyacentes, por ejemplo, si sólo se desea que exista conectividad entre los píxeles con intensidades 80,81 y 83, entonces V = {80,81,83}. Consideremos tres tipos básicos de conectividad:

Conectividad-4. Dos píxeles p y q con valores de V están 4-conectados si q está en el conjunto E_4 (p).

Conectividad-8. Dos píxeles p y q con valores de V están 8-conectados si q está en el conjunto E_8 (p).

Conectividad-m (mixta). Dos píxeles p y q con valores de V están m-conectados si q está en el conjunto E_4 (p) o q está en E_D (p) y E_4 (p)∩E_4 (q)=∅.

Un píxel p es contiguo a otro píxel q si están conectados. Se puede definir la adyacencia-4, 8 o m, dependiendo del tipo de conectividad especificada. Dos subconjuntos imagen S1 y S2 son contiguos si algún píxel se S1 es contiguo a algún píxel de S2.

Un camino desde el píxel p con coordenadas (x, y) hasta un píxel q con coordenadas (s, t) es una secuencia de varios píxeles con coordenadas,

$$(x\_0,y\_0),(x\_1,y\_1),\ldots,(x\_n, y\_n)$$

donde (x_0, y_0) = (x, y) y (s, t),(x_i, y_i) son adyacentes a (x_(i-1), y_(i-1)), con $1 \leq i \leq n$, y n es la longitud del camino. Se pueden definir caminos -4, 8 o m dependiendo del tipo de adyacencia usada.

Si p y q son píxeles de un subconjunto de imagen S, entonces p está conectado a q en S si existe un camino desde p hasta q formado de píxeles pertenecientes a S. Dado un píxel p cualquiera de S, el conjunto de píxeles de S que están conectados a p se llaman componente conectado de S. Se deduce que dos píxeles cualesquiera de un componente conectado están a su vez conectados entre sí y que los componentes conectados distintos son disjuntos.

Un camino simple es un camino sin píxeles repetidos y un camino cerrado es un camino simple en el cual el primer píxel es un vecino del último.

2.2.13. OPERACIONES DE FILTRADO

Las operaciones de filtrado basan su operatividad en la Convolución de la imagen utilizando el denominado núcleo de Convolución. Cabe distinguir dos tipos de filtros: paso alto y paso bajo, que en el contexto de la teoría de señales supone que los primeros dejan pasar las altas frecuencias de la señal y los segundos, las bajas. En el caso de las imágenes nos referimos a frecuencias espaciales. De una forma, las altas frecuencias se asocian a cambios bruscos de intensidad en pequeños intervalos espaciales, es decir, bordes, mientras que las bajas frecuencias se refieren a cambios lentos en la intensidad.

2.2.13.1. Filtros Paso Bajo

En la siguiente figura se muestran algunos núcleos de convolución que caracterizan los filtros paso bajo:

$$PB_1 \equiv \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$PB_2 \equiv \frac{1}{10}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$PB_3 \equiv \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

FIGURA 13: TRES NÚCLEOS REPRESENTATIVOS DE FILTROS PASO BAJO

2.2.13.2. Filtros Paso Alto

En la siguiente figura se muestran algunos núcleos de Convolución que caracterizan los filtros paso alto:

$$PA_1 \equiv \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$PA_2 \equiv \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$PA_3 \equiv \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

FIGURA 14: TRES NÚCLEOS REPRESENTATIVOS DE FILTROS PASO ALTO

2.2.14. HISTOGRAMA

El histograma de una imagen es la representación gráfica o analítica de la distribución relativa de cada valor posible de pixel de imagen, y en caso de imágenes grises de 8 bits será un vector de 256 componentes, siendo la componente i el número de pixeles de nivel i en la imagen, dividido por el número total de pixeles:

El histograma es formalmente la función estadística de densidad de probabilidad en forma discreta de los distintos niveles de gris dentro de la imagen.

$$histograma[i] = \frac{N_{pixeles\ \ de\ \ nivel\ \ i}}{N_{total}} \qquad 0 \le i \le 255$$

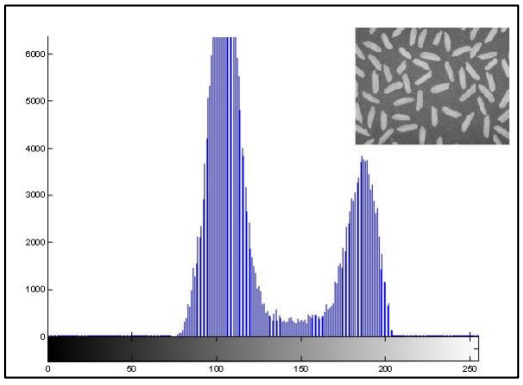

FIGURA 15 HISTOGRAMA

2.2.15. OPERACIONES MORFOLÓGICAS.

Las operaciones morfológicas a imágenes se definen como procedimientos en los cuales cada nuevo pixel de la imagen resultante es obtenido de una operación no lineal entre un conjunto de puntos de la imagen original F[x, y] y un conjunto de puntos conocido con el nombre de elemento estructurante S[x, y]. Este elemento estructurante recorre toda la imagen para obtener todos los puntos de la nueva imagen, sin cambiar el tamaño de la imagen de salida.

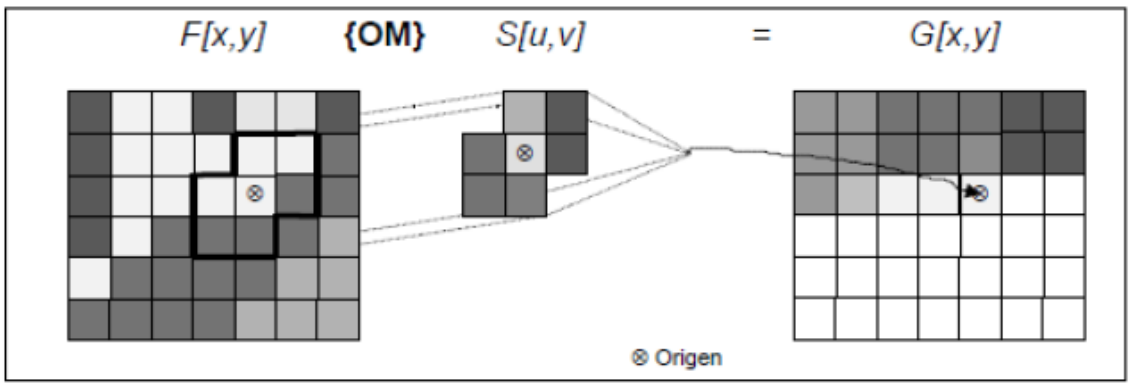

FIGURA 16 OPERACIÓN MORFOLÓGICA DE UN ELEMENTO ESTRUCTURANTE S[U,V] SOBRE UN ÁREA DE LA IMAGEN F[X,Y] PARA OBTENER UNA IMAGEN G[X,Y].

Dependiendo de los elementos estructurantes y de las operaciones utilizadas, los filtros morfológicos, pueden detectar bordes en las imágenes, filtrar objetos de tamaños menores a uno determinado, suavizar fondos de texturizados, detectar fallos en patrones de textura, etc.

2.2.16. Dilatación y Erosión.

Las operaciones morfológicas binarias de dilatación y erosión para imágenes binarias consisten en tomar el elemento estructurante (uno de cuyos puntos se considera el origen) y superponerlo a la imagen.

Si en el proceso de erosión el elemento estructurante está completamente contenido en la imagen, el punto de ésta se mantiene, de lo contrario pasa a ser del color del fondo. La erosión se aplica para reducir salientes de objetos, eliminar partes de tamaño menor que el elemento estructurante, eliminar conexiones insignificantes entre objetos mayores.

En el proceso de dilatación si parte del elemento estructurante es igual al contenido de la

imagen, el punto de la imagen pasa a ser igual al punto origen del elemento estructurante, de otra forma pasa a ser del color del fondo. La dilatación se aplica para rellenar entrantes de objetos y unir pequeñas separaciones en las imágenes.

2.2.17. EROSION BINARIA

La transformación de la erosión es el resultado de comprobar si el elemento estructurante B está completamente incluido dentro del conjunto X. Cuando no ocurre, el resultado de la erosión es el conjunto vacío:

$$\varepsilon_B(X) = X \Theta B = \{x \mid B_x \subseteq X\}$$

Cuando los objetos de la escena sean menores que el elemento estructurante, éstos desaparecerán. Otra interpretación de la erosión supone tomar el valor mínimo de la imagen en el entorno de vecindad definido por el elemento estructurante.

Su utilidad consiste en definir una geometría determinada al elemento estructurante y pasarlo sobre la imagen. Los objetos menores al elemento estructurante no aparecerán en la imagen resultante. Los objetos que queden de la transformación habrán sido degradados. Por tanto, la erosión supone una

degradación de la imagen. La aplicación iterativa de esta transformación hará que se eliminen todos los objetos existentes en la imagen.

2.2.18. DILATACIÓN BINARIA

La dilatación es la transformación dual a la erosión. El resultado de la dilatación es el conjunto de elementos tal que al menos algún elemento del conjunto estructurante B está contenido en el conjunto X, cuando B se desplaza sobre el conjunto X:

$$\delta_B(X) = X \oplus B = \{x \mid X \cap B_x \neq \varnothing\}$$

Esta operación representa un crecimiento progresivo del conjunto X. Al pasar el elemento estructurante dentro del conjunto, éste no se modificará. Sin embargo, en la frontera del conjunto X, al desplazar a B, el conjunto resultado se expansionará. La aplicación iterada de este operador haría degradar la imagen, haciendo coincidir el conjunto dilatado con la imagen.

La dilatación también se interpreta como el valor máximo del entorno de vecindad definido por el elemento estructurante.

Las aplicaciones de las operaciones de erosión seguida con una dilatación no son conmutativas. Los resultados son diferentes dando paso a las aperturas y cierres morfológicos.

2.2.19. ESPACIOS DE COLORES Y COLORIMETRÍA

Como ya se mencionó en una sección anterior, es posible representar los colores en base a una composición de colores fundamentales, imitando lo que hace nuestro ojo. En base a este hecho, es que se han creado espacios de colores que, de alguna manera u otra, permiten representar un conjunto de valores. Normalmente no se pueden representar todos los colores que son realmente

visibles, sin embargo, son lo suficiente como para representar la gran mayoría de colores que vemos regularmente.

Un espacio de color define un modelo de composición del color. Por lo general un espacio de color lo define una base de N vectores (por ejemplo, el espacio RGB lo forman 3 vectores: Rojo, Verde y Azul), cuya combinación lineal genera todo el espacio de color. Los espacios de color más generales intentan englobar la mayor cantidad posible de los colores visibles por el ojo humano, aunque existen espacios de color que intentan aislar tan solo un subconjunto de ellos.

Existen espacios de color de:

Una dimensión: escala de grises, escala Jet, etc.

Dos dimensiones: sub-espacio rg, sub-espacio xy, etc.

Tres dimensiones: espacio RGB, HSV, YCbCr, YUV, YI'Q', etc.

Cuatro dimensiones: espacio CMYK.

De los cuales, los espacios de color de tres dimensiones son los más extendidos y los más utilizados para la mayoría de las aplicaciones. En el caso que se cuente con un espacio de una dimensión de profundidad, estaríamos en presencia de un espacio de color que en la práctica puede representar el grado de luminosidad de la imagen; es básicamente una imagen en escala de grises, y no entrega información de color. Pero si queremos trabajar con colores, los modelos existentes especifican tres coordenadas, o atributos, que representan su posición dentro de un espacio de color específico. Estas coordenadas no nos dicen cuál es el color, sino que muestran dónde se encuentra un color dentro de un espacio de color en particular. A continuación, se hará una descripción de los espacios de color más relevantes, así como los más importantes para este trabajo.

2.2.19.1. Espacio de color RGB

Este modelo es el más utilizado para visualizar imágenes digitales en una pantalla en los formatos actuales. Como se ha dicho anteriormente, es un modelo aditivo, en el que sumando distintas cantidades de colores primarios podemos conseguir otros colores. En los archivos gráficos que utilizan este modelo utilizamos para representar una tripleta (R, G, B).

Contienen tres planos de imágenes independientes, uno por cada color primario. Cuando estos planos se inyectan a un monitor que utiliza este sistema, la pantalla de fósforo reproduce una imagen a color. Es un sistema aditivo que, variando la cantidad de color rojo, verde y azul agregados al negro, se produce nuevos colores. En los archivos gráficos el sistema RGB se usa para representar cada píxel con una tripleta o terna de la forma (R, G, B), quedando representado en un sistema cartesiano, como se puede apreciar en la figura.

En este sistema los colores vienen definidos por un punto del cubo, así los vértices comunes son los colores secundarios, el origen de coordenadas es el negro, y el punto donde se sumarian los 3 colores primarios será el vértice del color blanco. Y si trazamos una línea que una el negro con el blanco representará toda la escala de grises.

A un píxel se le asigna un valor entre 0(negro) y 255(blanco). La idea es la siguiente: si por ejemplo queremos un color rojo tendríamos que darle un valor alto a la coordenada del rojo y valores más pequeños a las otras dos coordenadas. Si todas las coordenadas son 255 tendremos un blanco puro e igualmente si todas son 0 tendremos un negro. Y si las 3 son iguales entonces tendremos un gris.

Las imágenes que utilizan este modelo pueden reproducir en la pantalla hasta 16.7 millones de colores, ya que tendremos 3 canales (que presentan a cada uno de los planos) y 8 bits para presentar 255 valores, es decir tendremos 3x8 bits de información de color para cada píxel.

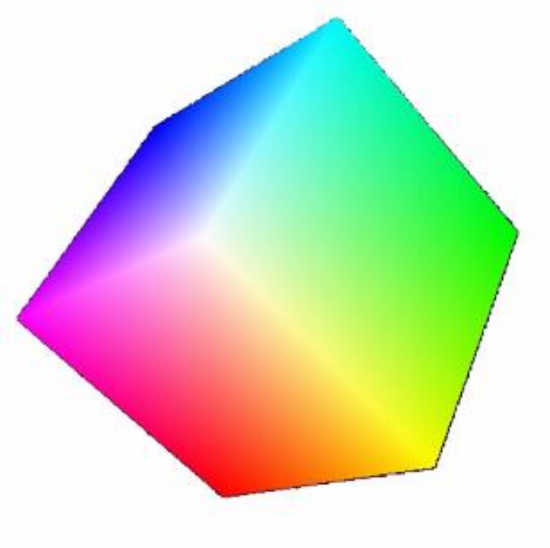

FIGURA 17 MODELO ADITIVO RGB

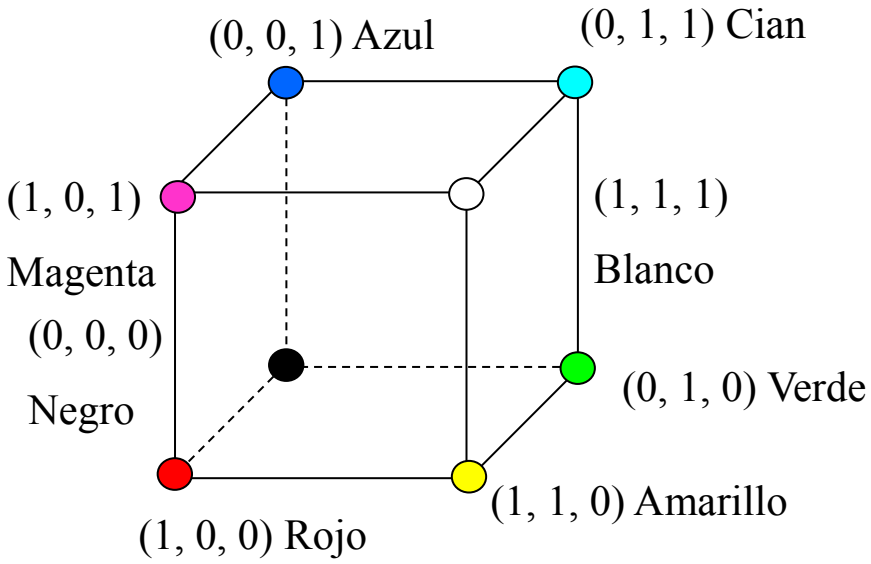

FIGURA 18 PESO DE LAS 3 COMPONENTES EN EL MODELO RGB

2.2.19.2. Espacio de color HSV

El modelo HSV (del inglés Hue, Saturation, Value) (Tonalidad, Saturación, Valor), también llamado HSB (Hue, Saturation, Brightness) (Tonalidad, Saturación, Brillo), define un modelo de color en términos de sus componentes constituyentes en coordenadas cilíndricas:

Tonalidad, el tipo de color (como rojo, azul o amarillo). Se representa como un grado de ángulo cuyos valores posibles van de 0 a 360º (aunque para algunas aplicaciones se normalizan del 0 al 100 %). Cada valor corresponde a un color. Ejemplos: 0 es rojo, 60 es amarillo y 120 es verde.

Saturación. Se representa como la distancia al eje de brillo negro-blanco. Los valores posibles van del 0 al 100 %. A este parámetro también se le suele llamar "pureza" por la analogía con la pureza de excitación y la pureza colorimétrica de la colorimetría. Cuanto menor sea la saturación de un color, mayor tonalidad grisácea habrá y más decolorado estará. Por eso es útil definir la insaturación como la inversa cualitativa de la saturación.

Valor del color, el brillo del color. Representa la altura en el eje blanco negro. Los valores posibles van del 0 al 100 %. 0 siempre es negro. Dependiendo de la saturación, 100 podría ser blanco o un color más o menos saturado. La figura muestra una representación del espacio HSV.

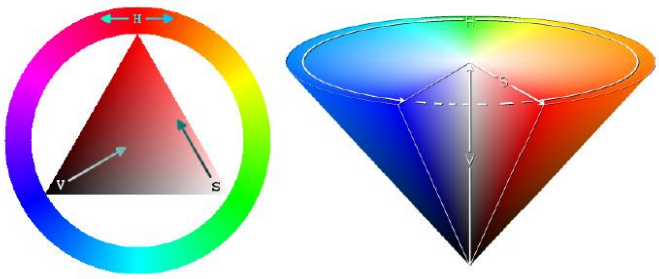

FIGURA 19 MODELO HSV

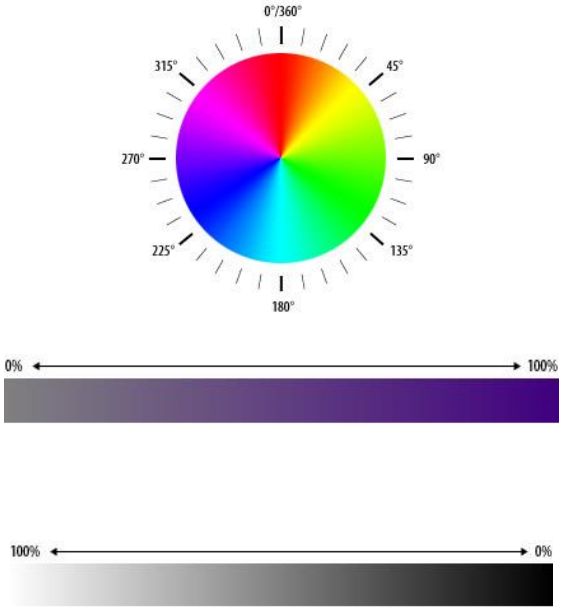

FIGURA 20 TINTE, SATURACIÓN Y VALOR

2.2.20. MOMENTOS INVARIANTES DE HU

Los momentos invariantes fueron propuestos por primera vez por Hu. Estos momentos pueden ser considerados como un promedio ponderado de los pixeles de una imagen. Hu computo sus invariantes utilizando los momentos geométricos, los cuales son variantes a la rotación y al escalamiento. Los momentos geométricos se definen como:

$$\mu_{pq} = \iint (x - \bar{x})^p (y - \bar{y})^q f(x, y) dy dx$$

Donde μ_{pq} es el momento geométrico de orden (p+q), f(x; y) es el valor del pixel en la posición (x; y) de la imagen y (x; y) es el centroide de la misma.

Partiendo de estos momentos podemos obtener n_{pq}, un momento de orden (p + q) que es invariante al escalamiento:

$$n_{pq} = \frac{\mu_{pq}}{\mu_{00}^{1+\frac{p+q}{2}}}$$

Utilizando n_{pq}, Hu logro un conjunto de momentos invariantes, todos con un grado menor o igual a 3.

2.2.21. OpenCV[3]

OpenCV es una biblioteca multiplataforma que permite desarrollar aplicaciones de visión por ordenador en tiempo real. Se centra en el procesamiento de imágenes, captura de vídeo y análisis, incluyendo características como detección de rostros y detección de objetos.

OpenCV significa Open Source Computer Vision Library; por lo tanto, es una librería de tratamiento de imágenes, destinada principalmente a aplicaciones de visión por computador en tiempo real. Una de las ventajas principales es que puede funcionar en muchas plataformas, existen versiones para Windows, Linux y MacOs.

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel; desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

OpenCV es multiplataforma, existiendo versiones para GNU/Linux, Mac OSX y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estéreo y visión robótica.

[3] https://www.tutorialspoint.com/opencv/

FIGURA 21 LOGO OPENCV

2.2.21.1. Características principales

- Optimizado para procesamiento de imágenes en tiempo real y aplicaciones de visión artificial
- La interfaz primaria de OpenCV está en C + +
- También hay interfaces C, Python y JAVA completas
- Las aplicaciones OpenCV se ejecutan en Windows, Android, Linux, Mac e iOS
- Optimizado para procesadores Intel

2.2.21.2. Módulos OpenCV

OpenCV tiene una estructura modular. A continuación, se enumeran los módulos principales de OpenCV.

Núcleo

Este es el módulo básico de OpenCV. Incluye estructuras de datos básicas (por ejemplo, estructura de datos Mat) y funciones básicas de procesamiento de imágenes. Este módulo también es ampliamente utilizado por otros módulos como highgui, etc.

Highgui

Este módulo ofrece funciones sencillas de interfaz de usuario, varios códecs de imagen y vídeo, capacidades de captura de imagen y video, manipulación de

ventanas de imagen, manejo de barras de seguimiento y eventos de ratón y etc. Si desea capacidades de interfaz de usuario avanzadas, debe utilizar marcos de interfaz de usuario como Qt, WinForms, etc, por ejemplo, cargar y mostrar imágenes, capturar vídeo desde un archivo o cámara, escribir imagen y vídeo en el archivo

Imgproc

Este módulo incluye algoritmos básicos de procesamiento de imágenes incluyendo filtrado de imágenes, transformaciones de imagen, conversiones de espacio de color y etc.

Vídeo

Este es un módulo de análisis de video que incluye algoritmos de seguimiento de objetos, algoritmos de substracción de fondo y etc.

Objdetect

Esto incluye algoritmos de detección y reconocimiento de objetos para objetos estándar.

OpenCV ahora se utiliza extensivamente para desarrollar el procesamiento avanzado de la imagen y las aplicaciones de la visión de la computadora. Ha sido una herramienta para estudiantes, ingenieros e investigadores en todos los rincones del mundo.

2.3. GLOSARIO DE TÉRMINOS BÁSICOS

ESTADÍSTICA: es una rama de las matemáticas y una herramienta que estudia usos y análisis provenientes de una muestra representativa de datos, que busca explicar las correlaciones y dependencias de un fenómeno físico o natural, de ocurrencia en forma aleatoria o condicional.

Es transversal a una amplia variedad de disciplinas, desde la física hasta las ciencias sociales, desde las ciencias de la salud hasta el control de calidad. Además, se usa en áreas de negocios o instituciones gubernamentales ya que su principal objetivo es describir al conjunto de datos obtenidos para la toma de decisiones o bien, para realizar generalizaciones sobre las características observadas.

En la actualidad, la estadística es una ciencia que se encarga de estudiar una determinada población por medio de la recolección, recopilación e interpretación

de datos. Del mismo modo, también es considerada una técnica especial apta para el estudio cuantitativo de los fenómenos de masa o colectivo.

REGRESIÓN: es un proceso estadístico para estimar las relaciones entre variables. Incluye muchas técnicas para el modelado y análisis de diversas variables, cuando la atención se centra en la relación entre una variable dependiente y una o más variables independientes (o predictoras). Más específicamente, el análisis de regresión ayuda a entender cómo el valor de la variable dependiente varía al cambiar el valor de una de las variables independientes, manteniendo el valor de las otras variables independientes fijas. Más comúnmente, el análisis de regresión estima la esperanza condicional de la variable dependiente dadas las variables independientes - es decir, el valor promedio de la variable dependiente cuando se fijan las variables independientes.

SISTEMA: Un sistema es un conjunto de elementos relacionados entre sí que funciona como un todo. La palabra sistema procede del latín systēma, identificado en español como "unión de cosas de manera organizada". De esta palabra se derivan otras como antisistema o ecosistema. Los elementos que componen un sistema pueden ser variados, como una serie de principios o reglas estructuradas sobre una materia o teoría.

LOCALIZACIÓN DE OBJETOS EN IMÁGENES: Cuando queremos localizar un objeto en una imagen, una técnica utilizada consiste en recorrer toda la imagen aplicando múltiples ventanas detectoras de diferentes tamaños. Para cada ventana se aplica algún clasificador que nos indica si existe el objeto que intentamos. Al ser un algoritmo que se basa en diversos recorridos de la imagen, el desplazamiento de la ventana a través de ella es un factor muy importante. Con la disminución del desplazamiento de la ventana, el tiempo de proceso crece exponencialmente. Por otro lado, un desplazamiento demasiado grande podría comportar la no detección de objetos por no quedar localizados dentro de su ventana.

BLOB ANÁLISIS: Un algoritmo utilizado en visión artificial que identifica objetos segmentados y los mide según parámetros morfométricos (tamaño, diámetro, perímetro, etc.) o densitométicos (nivel de gris medio, media, color)

ATRIBUTO: Un atributo es una especificación que define una propiedad de un Objeto, elemento o archivo. También puede referirse o establecer el valor específico para una instancia determinada de los mismos.

CCD: es la abreviatura de Charge-Coupled Device. Un sensor CCD es un dispositivo semiconductor sensible a la luz, que convierte las partículas de luz (fotones) en cargas eléctricas (electrones). Las cámaras CCD son una de los dos tipos de cámaras que dominan el mercado de la visión, conjuntamente con las cámaras CMOS.

CMOS: Es el acrónimo de Complementary Metal Oxide Semiconductor. Esta tecnología funciona como un fotodiodo donde la luz genera una corriente que es representativo de la cantidad de luz que impacta en cada píxel, por tanto difiere significativamente de la tecnología CCD. Hay un número de ventajas en la utilización de los sensores CMOS con respecto a los CCD entre los que se destaca el coste, velocidad, anti-blooming, y características de respuesta programable (Ej. Respuesta con múltiples pendientes).

ESCALA DE GRISES: En visión este término se refiere a una imagen monocroma con una gradación de grises. Una cámara monocroma que trabaje a 8 bits generará una imagen con 256 tonos de gris. Si la cámara trabaja a 12 bits su escala de gises será de 4096 niveles.

HISTOGRAMA: Es una representación gráfica de los valores de todos los píxeles de la imagen. Generalmente a la izquierda de la gráfica se representa el valor cero correspondiente al negro y a la derecha se representa el valor máximo (que depende del número de bits) correspondiente al blanco. La curva del histograma representa la cantidad de píxeles de cada valor de gris que hay en la imagen. Si la imagen es en color se acostumbran a representar los histogramas de valores de rojo, verde y azul.

MODELO: Un modelo es una representación de un objeto, sistema o idea, de forma diferente al de la entidad misma. El propósito de los modelos es ayudarnos a explicar, entender o mejorar un sistema. Un modelo de un objeto puede ser una réplica exacta de éste o una abstracción de las propiedades dominantes del objeto.

INTELIGENCIA ARTIFICIAL: Es aquella inteligencia exhibida por artefactos creados por humanos (es decir, artificial). A menudo se aplica hipotéticamente a

los computadores. El nombre también se usa para referirse al campo de la investigación científica que intenta acercarse a la creación de tales sistemas.

MORFOLOGÍA MATEMÁTICA: es una teoría y técnica para el análisis y tratamiento de las estructuras geométricas, basada en la teoría de conjuntos, teoría de retículos, topología y funciones aleatorias. La morfología matemática es comúnmente aplicada más a las imágenes digitales, pero puede ser empleada también en gráficos, mallas poligonales, sólidos y muchas otras estructuras espaciales.

2.4. HIPÓTESIS

Utilizando visión artificial se predecirá y seguirá la posición de una esfera en tiempo real.

III. MARCO METODOLÓGICO

3.1. MÉTODOS Y PROCEDIMIENTOS

3.1.1. DESCRIPCIÓN DEL PROYECTO

Para el desarrollo del proyecto de tesis se identifica el recorrido realizado por una esfera dentro del escenario analizado. La predicción de posición consiste en utilizar la información correspondiente a los recorridos identificados junto con un modelo del movimiento de los objetos para predecir sus recorridos futuros.

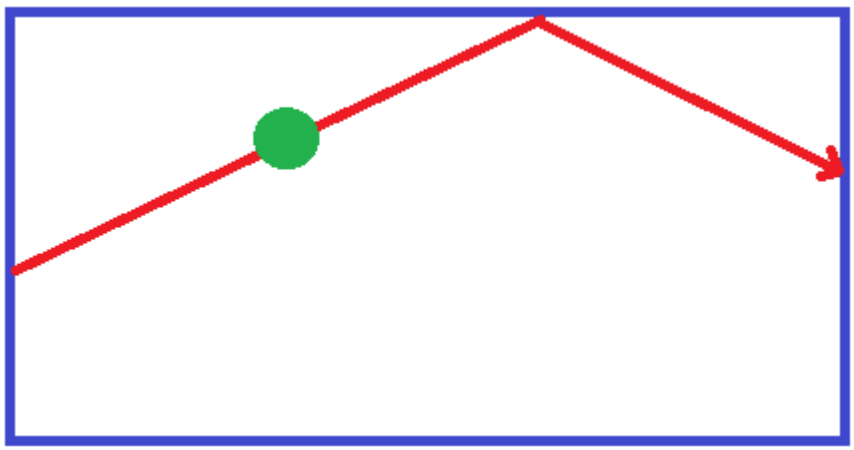

FIGURA 22 PREDICCIÓN DE POSICIÓN Y TRAYECTORIA DE UNA ESFERA

El sistema está conformado por los elementos mostrados en la figura siguiente:

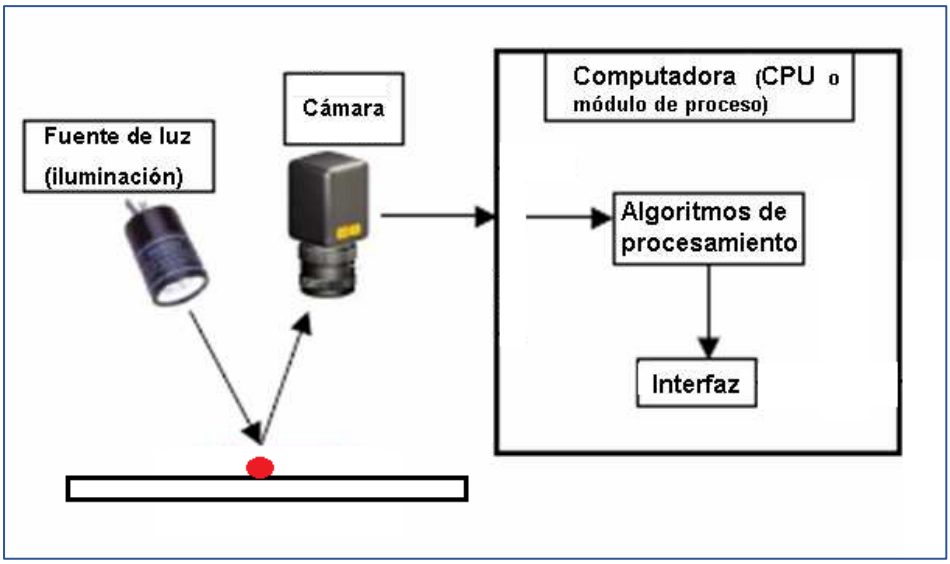

FIGURA 23 ELEMENTOS DEL SISTEMA

La información de movimiento de la esfera se obtiene por medio de una cámara de video la cual se encuentra fija en la parte superior del área de trabajo, capturando de manera secuencial, imágenes de todo lo sucedido en esta zona.

Esta información es enviada a un computador, el cual se encarga de procesar a una tasa de 30 cuadros por segundo, las imágenes tomadas y así determinar cuál será la ubicación final de la esfera.

3.1.2. HERRAMIENTAS DE DESARROLLO

En el desarrollo del proyecto se utilizó el lenguaje de programación Python, corriendo sobre el sistema operativo Linux. Se utilizaron las librerías de visión por Computadora OpenCv y Matlab 2015 para las simulaciones y modelamiento del sistema.

3.1.3. METODOLOGÍA

Mas allá de las variedades de situaciones y de las características propias de cada problema particular, en cualquier sistema de seguimiento y predicción de posición podemos identificar a grandes rasgos 4 etapas.

- La captura de datos.

- El proceso de identificación de la esfera.

- El armado de las trayectorias.

- El proceso de predicción de las posiciones futuras

Entonces esta será la metodología por utilizar para cumplir con el objetivo del proyecto, pero antes se hará un análisis con el software científico Matlab, utilizando datos artificiales.

Los datos ingresados al sistema constituyen una representación de la realidad sobre la cual se debe trabajar. La información de entrada está constituida por una secuencia de imágenes digitales (frames) capturadas a través de un sensor óptico (cámara). En la mayoría de los casos debe realizarse un preprocesamiento a los datos capturados con el fin de mejorar la calidad de los mismos. Cuando se trabaja con imágenes digitales existen numerosos problemas relacionados con la captura, entre los cuales podemos mencionar el ruido, malas condiciones de iluminación, distorsiones geométricas provocadas por la lente de la cámara o debido a la perspectiva, imagen borrosa o fuera de foco, etc. Es necesario aplicar a la imagen capturada un conjunto de algoritmos con el fin de obtener una nueva imagen que resulte más adecuada para el procesamiento posterior.

Podemos plantear el problema de predicción de posición de la siguiente manera:

Dada una secuencia de puntos P_1, P_2,....P_n que representa las coordenadas observadas en períodos regulares de tiempo del recorrido realizado por un objeto en movimiento, se pretende estimar m posiciones futuras P_{n+1} , P_{n+2},....P_{n+m} correspondiente a los m instantes de tiempo futuro t_{n+1}, t_{n+2}...t_{n+m}.

Las coordenadas de la trayectoria del objeto fueron obtenidas en los primeros pasos del sistema y sus cálculos están influenciados por la presencia de ruido.

Con el objetivo de modelar el movimiento del objeto en función del tiempo, lo descomponemos en dos funciones, posX(t) y posY(t), que representan el movimiento en cada uno de los ejes. Las funciones posX(t) y posY(t) indican las coordenadas ocupada por el objeto en los ejes X e Y respectivamente, en el instante de tiempo t.

Cuando el movimiento de un objeto es rectilíneo uniforme, su desplazamiento en cada uno de los ejes es una función lineal del tiempo. El problema de predicción

de la trayectoria consiste en estimar los valores que tomarán posX(t) y posY(t) en instantes de tiempo posteriores.

La técnica de regresión lineal permite obtener una buena aproximación de las rectas posX(t) y posY(t) a partir de las observaciones ruidosas.

Sean (t_1,c_1), (t_2,c_2), ..., (t_n,c_n) los pares (tiempo, coordenada). Deseamos armar una función lineal de la forma:

$$C = b_0 + b_1 T$$

Las siguientes fórmulas obtienen los parámetros b_0 y b_1 que ajustan la recta con el menor error posible a la muestra.

$$b_1 = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}$$

$$b_0 = \frac{\sum_{i=1}^{n} y_i - b_1 \sum_{i=1}^{n} x_i}{n}$$

Armando las rectas de regresión posX(t) y posY(t) tenemos una estimación del movimiento del objeto en cada uno de los ejes. Estas rectas permiten estimar la posición que ocupará el objeto en instantes de tiempo posteriores a los analizados.

Para comprender mejor esto, se hace una simulación en Matlab con los siguientes datos:

x	8	2	11	6	5	4	12	9	6	1
y	3	10	3	6	8	12	1	4	9	14

Haciendo el código en Matlab:

```
x = [8 2 11 6 5 4 12 9 6 1];

y = [3 10 3 6 8 12 1 4 9 14];
```

```
plot(x,y,'xr')
xlabel('Valores de X');
ylabel('Valores de Y');
grid
axis([-2 14 -2 16])
```

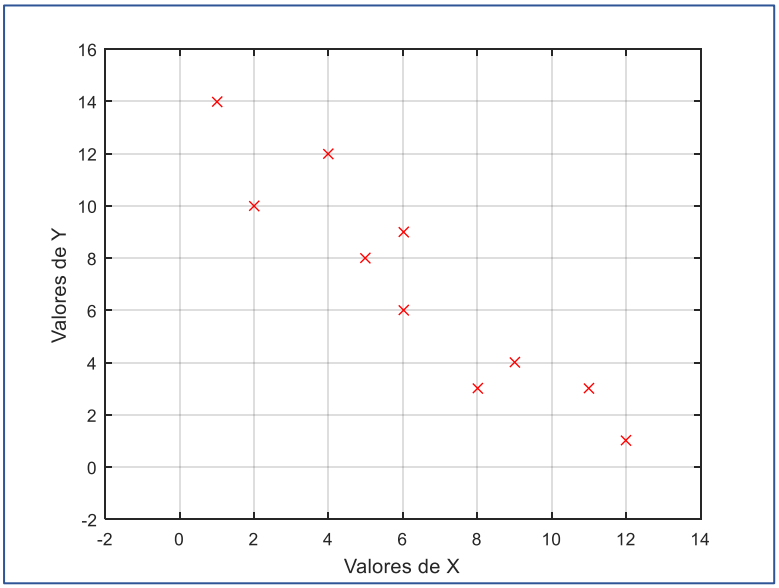

FIGURA 24 COORDENADAS DE PARES ORDENADOS

Utilizando Mínimos cuadrados obtenemos la ecuación de la recta que mejor se ajusta a los datos:

```
x = [8 2 11 6 5 4 12 9 6 1];
y = [3 10 3 6 8 12 1 4 9 14];
plot(x,y,'xr')
xlabel('Valores de X');
ylabel('Valores de Y');
grid
axis([-2 14 -2 16])
hold on
```

```
% Hallamos la pendiente (m):

n = length(x)

sx = sum(x)

sy = sum(y)

sxy = sum(x.*y)

sxc = sum(x.^2)

m = (sxy - (sx*sy/n)) / (sxc-(sx^2/n))

% para calcular la intercepción en y (b):

xp = sx/n

yp = sy/n

b = yp - m*xp

% Ecuacion de la recta es:

nx = 0:12;

ny = m*nx + b

plot(nx,ny,'b')

hold off
```

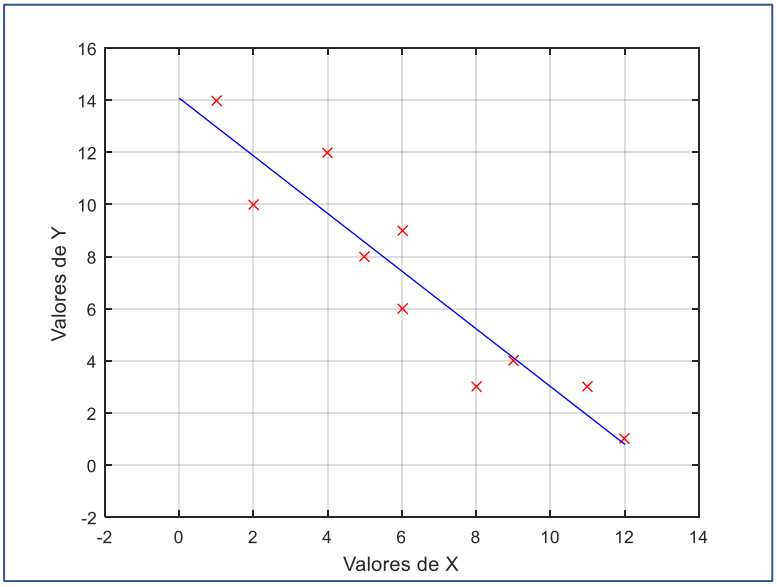

FIGURA 25 RECTA QUE DESCRIBE LA POSICIÓN

Ahora como ya podemos hallar la ecuación de la recta, para diferentes valores de pares ordenados en instantes previos de t, podemos predecir, el valor de y en cualquier instante (para una pendiente m=0.20 y b=15):

```
lz=0;

ld=50;

ls=30;

lf=0;

line([lz ld ld ld ld lz lz lz],[lf lf lf ls ls ls ls lf])

axis([lz-5 ld+5 lf-5 ls+5])

xlabel('Valores de X');

ylabel('Valores de Y');

grid

hold on

% Ecuacion de la recta es:

m=0.20; b=15;

for nx=lz:0.5:ld

   ny = m*nx + b;

   plot(nx,ny,'xr')

end

hold off
```

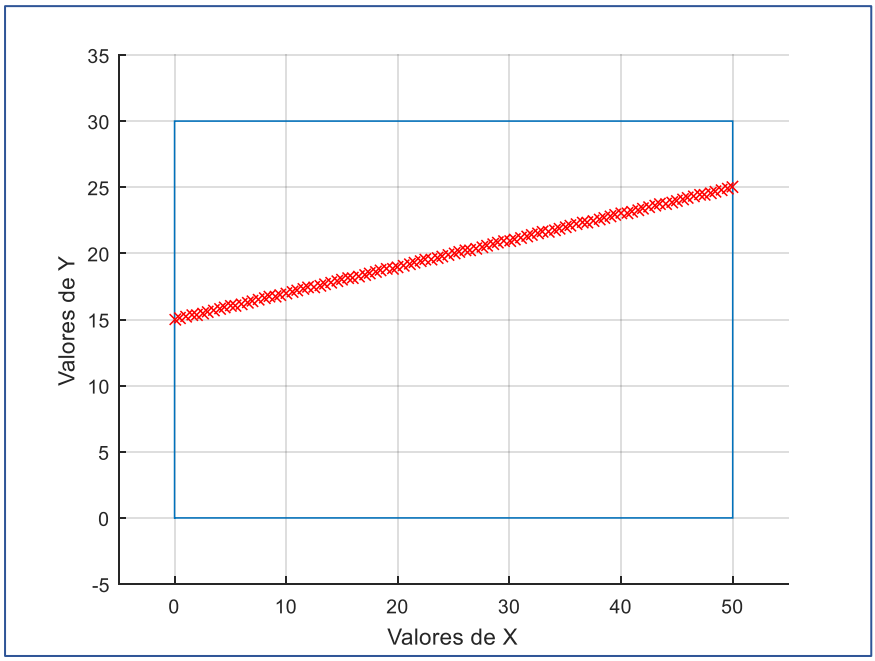

FIGURA 26 ***RECTA CON PENDIENTE m = 0.2***

para una pendiente m=0.40

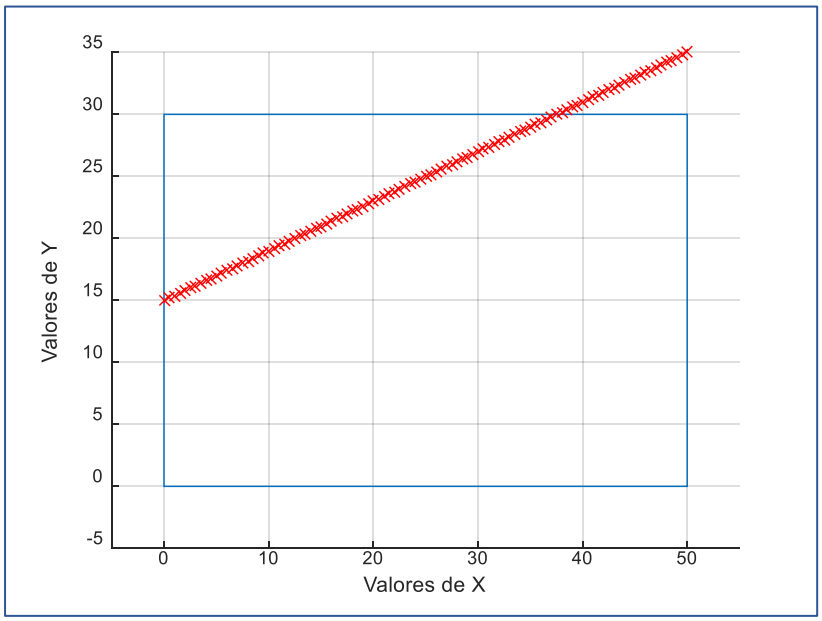

FIGURA 27 ***RECTA CON PENDIENTE m = 0.4***

Observamos en la figura anterior que existe la posibilidad de que la esfera rebote, entonces, hacemos un cambio de pendiente:

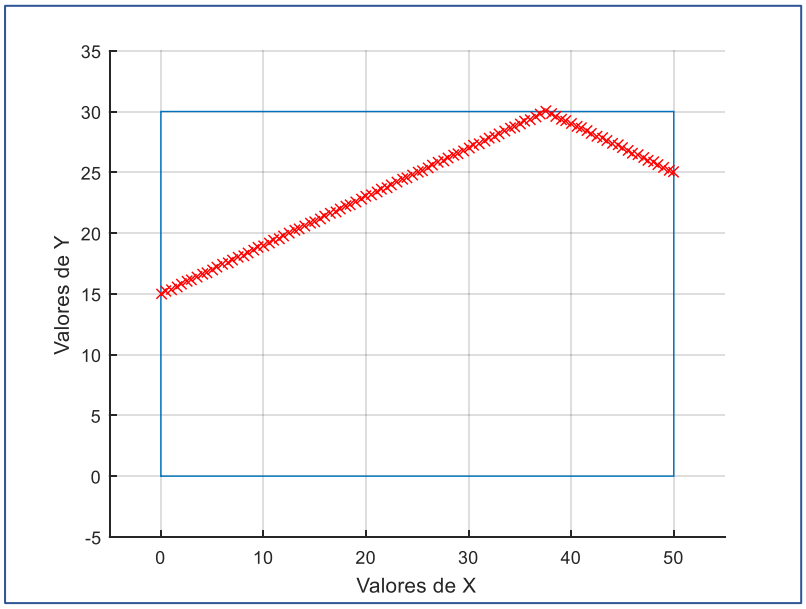

FIGURA 28 RECTA CON PENDIENTE m = 0.4 CON REBOTE

para una pendiente m=0.60

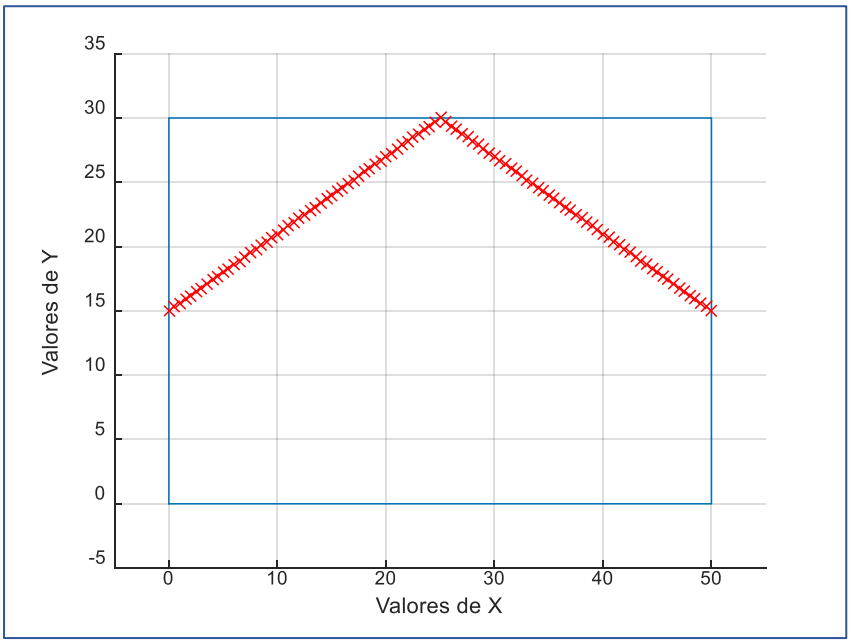

FIGURA 29 RECTA CON PENDIENTE m = 0.6

para una pendiente m=1.60

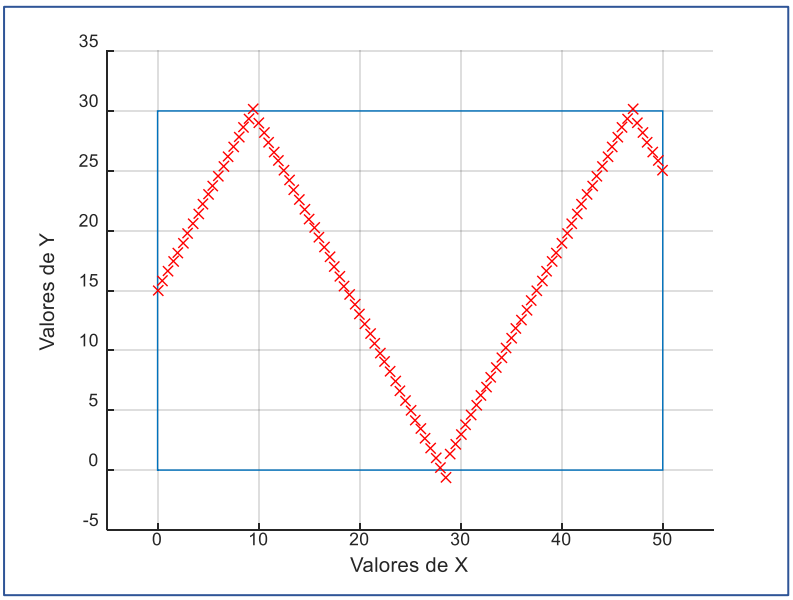

FIGURA 30 RECTA CON PENDIENTE m = 1.6 CON REBOTES

para una pendiente m=-1.20

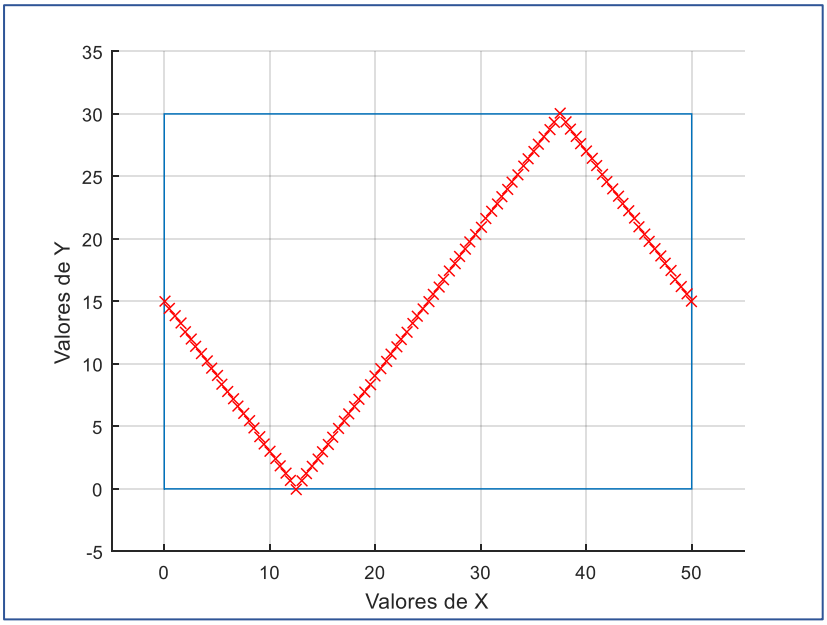

FIGURA 31 RECTA CON PENDIENTE m = -1.2 CON REBOTES

3.1.3.1. ACONDICONAMIENTO DE LA ESCENA

Todo sistema de visión artificial se lleva a cabo acondicionando la escena, como dijimos en la parte de delimitación, el sistema se hará teniendo en cuenta una iluminación natural.

La cámara debe colocarse con un soporte paralelo al plano y a una cierta distancia fija del área de recorrido de la esfera.

3.1.3.2. ADQUISICIÓN DE VIDEO

La toma de las secuencias de video, se hacen en el formato RGB. Las imágenes son de tamaño 1280 pixeles de ancho, por 740 pixeles de alto, que nos daría una resolución de 1 Megapíxel.

La velocidad de los fotogramas con que se trabajo fue de 30 fotogramas por segundo. Pero cabe recalcar que hasta aquí no se está tomando en cuenta el tiempo de procesamiento entre fotograma y fotograma.

Para comenzar a trabajar con procesamiento de imágenes en el lenguaje de programación Python, se utiliza la librería OpenCV.

Lo primero que hay que hacer es importar la librería:

```
import cv2
```

Creamos una variable de camara y asignamos la cámara webcam disponible con conexión usb instalada en la PC con el valor de "1"

```
cap = cv2.VideoCapture(1)
```

luego dentro de un bucle infinito hay que leer una imagen o fotograma del video que se está capturando:

```
frame = cap.read()
```

Entonces cada frame o fotograma que se lee, se procesa.

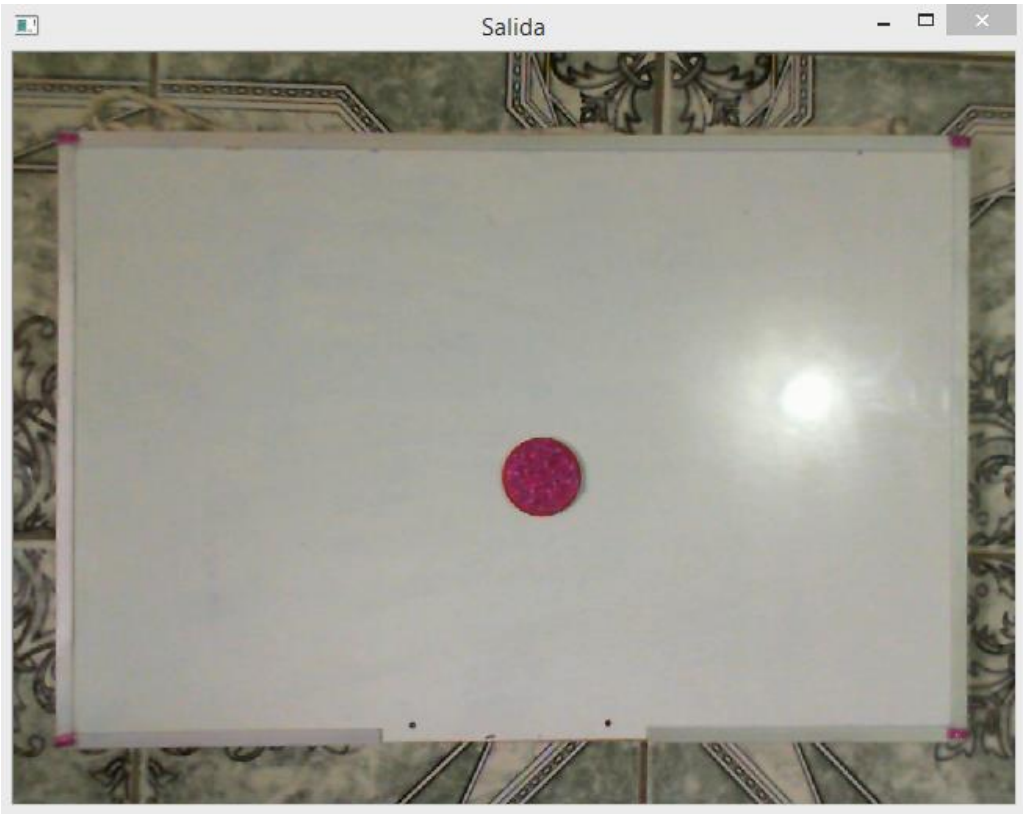

FIGURA 32 CAPTURA DE VIDEO DE LA ESCENA

3.1.3.3. CONVERSIÓN A MODELO HSV Y BINARIZACIÓN

Luego de leer un fotograma, el siguiente paso es detectar el área de trabajo y la esfera. Para ambos casos se trabajó con señalizaciones de un determinado color, que fue el fucsia.

Para segmentar el área de trabajo se recurrió ha poner en las 4 esquinas, 4 trozos de microporoso de color fucsia, de 10mm x 6 mm, como se puede apreciar en la anterior figura.

Para segmentar la esfera se recurrió, también, a poner 1 trozo de microporoso de color fucsia, de diámetro 50mm, como también se observa en la figura anterior.

La conversión del modelo RGB a HSV se hace con la siguiente función:

cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

Esta función devuelve las 3 componentes en el modelo HSV: Tono, Saturación y Brillo. Cada componente se encuentra por separado.

Los valores máximo y mínimo de Tono (H), Saturación (S) y Brillo (V) se obtuvieron de acuerdo con el color fucsia y estos son:

Hmin = 132

Hmax = 256

Smin = 67

Smax = 256

Vmin = 40

Vmax = 256

Los valores se encuentran normalizados entre 1 y 256. Como sabemos el Tono (H) se mide en un ángulo de 0° a 360°, por lo tanto, 0° se corresponde con el valor 1 y 360° se corresponde con el valor digital 256.

Hmin = 132

Hmax = 256

Hallando Hmin:

Hmin = (186 x 256) / 360 = 132

Hmax = (360 x 256) / 360 = 256

Para el caso de la Saturación, el grado de pureza del color, los valores son del 0 al 100%. 0 se corresponde con 0% y 256 se corresponde con 100%:

Smin = 67

Smax = 256

Smin = (26 x 256) / 100 = 67

Smax = (100 x 256) / 100 = 256

Para el Brillo sucede lo contrario, 0% se corresponde con 256 (grado de negro que tiene el color) y 100% se corresponde con 0 (grado de blanco que tiene el color):

Vmin = 40

Vmax = 256

Vmin = (15 x 256) / 100 = 40

Vmax = (100 x 256) / 100 = 256

Estos son los valores para los cuales se podrá segmentar los objetos de color fucsia, del resto de la imagen.

Estos valores los utilizamos con la siguiente función:

```
cv2.inRange(hsv, np.array((Hmin,Smin,Vmin)), np.array((Hmax,Vmax,Smax)))
```

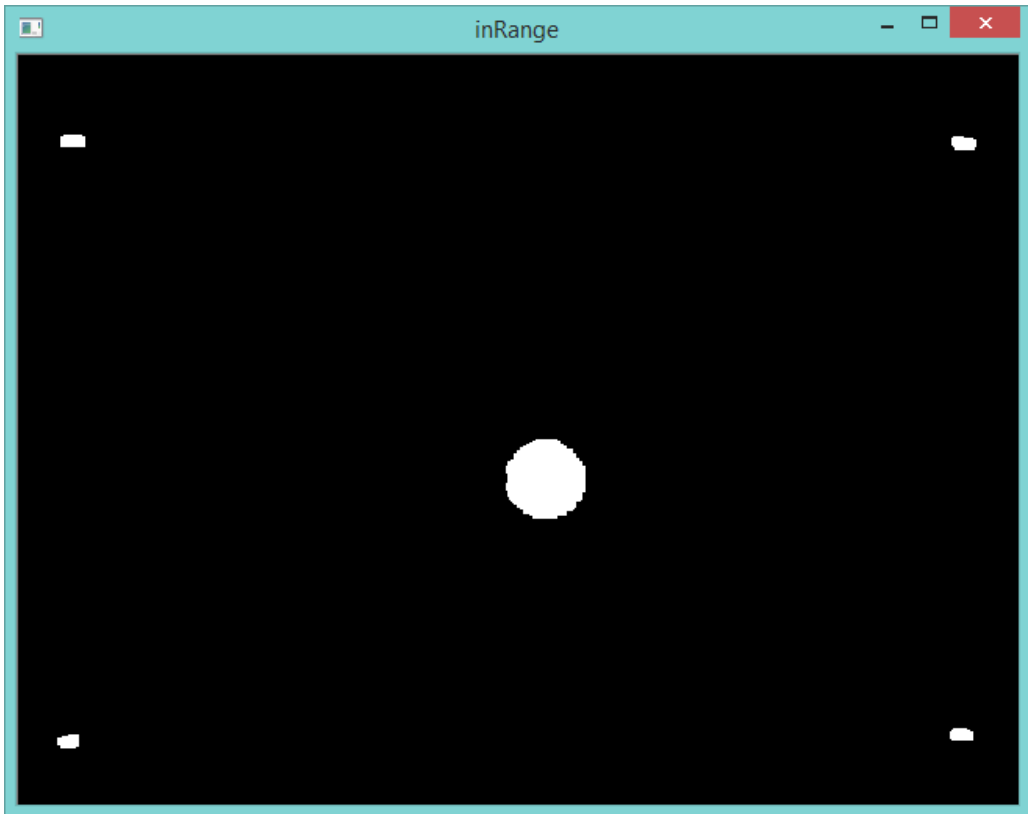

FIGURA 33 IMAGEN BINARIZADA

A los objetos les asignamos el color blanco y al fondo le asignamos color negro. En este proceso nos quedarnos con una imagen en blanco y negro y se llama binarización.

3.1.3.4. ELIMINACIÓN DE RUIDO

En la escena, ahora tendríamos que quedarnos con 5 objetos blancos sobre un fondo negro. De hecho, que se espera un poco de ruido en la imagen y para eliminarlos se erosiona la imagen con un elemento estructurante de 3 x 3 y luego se dilata la imagen con elemento estructurante también rectangular de 5 x 5. El resultado es la imagen binaria con 5 objetos.

Las funciones son las siguientes:

```
cv2.erode (bn_img,cv2.getStructuringElement(cv2.MORPH_RECT,(3,3)),iterations = 1)
cv2.dilate (bn_img,cv2.getStructuringElement(cv2.MORPH_RECT,(5,5)),iterations = 1)
```

3.1.3.1. DETECCIÓN Y SEGMENTACIÓN DE OBJETOS

Ahora que tenemos 5 objetos blancos sobre fondo negro, tenemos que detectar cuál es la esfera y cuál son los objetos de la esquina.

Para ello, nos basamos en el área de cada objeto, siendo la esfera la de mayor tamaño, mayor área. Si el área está entre 20 y 150 pixeles, se trata de los 4 objetos de las esquinas y si el área es mayor a 200 pixeles se trata de las esferas. Los centroides de cada objeto se obtienen en base a los momentos.

```
area = cv2.contourArea(c)
M = cv2.moments(c)
cx = int(M['m10']/M['m00'])
cy = int(M['m01']/M['m00'])
if area>20 and area<150:
   if(def_area == 0):
      cv2.circle(frame,(cx,cy),10,(255,255,0), 2)
      vX[num] = cx
      vY[num] = cy
```

```
            num = num + 1
            if(num > 3):
                num = 0
    if area>200:
        esfX = cx
        esfY = cy
```

Mostramos un circulo verde en la posición en la que se encuentra el objeto

```
        cv2.circle (frame,(cx,cy),20,(0,255,0), 2)
```

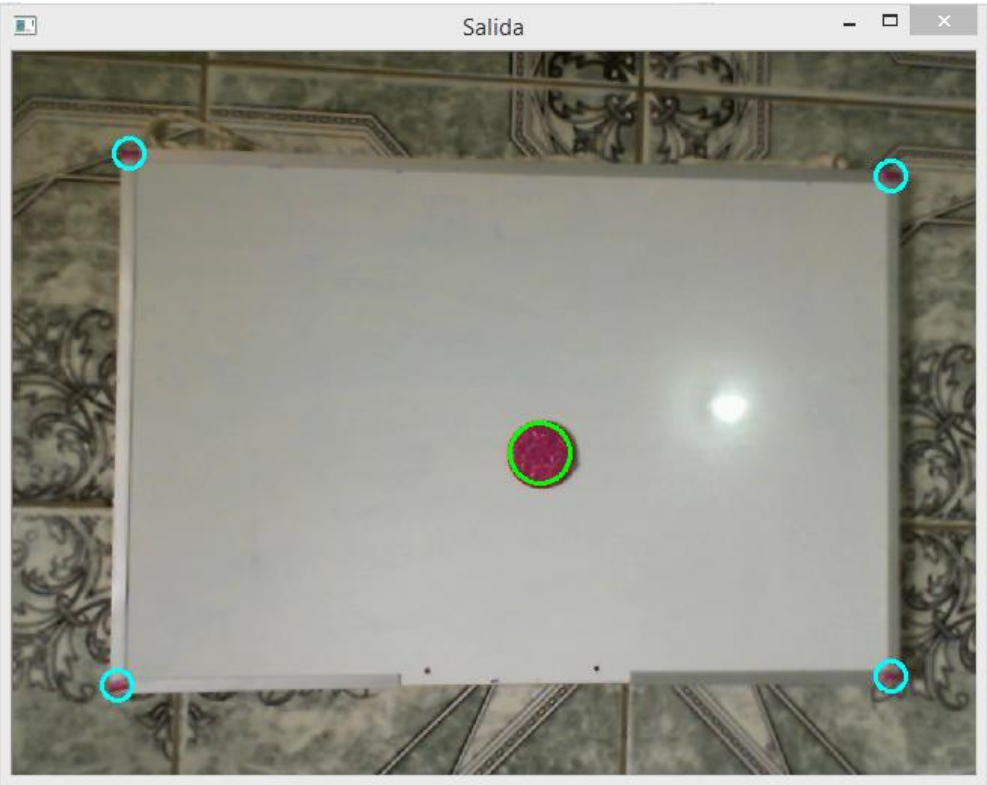

FIGURA 34 LOCALIZACIÓN DE OBJETOS

En base al área de cada objeto podemos observar las esquinas detectadas. Sus coordenadas (x, y) delimitan el área de trabajo.

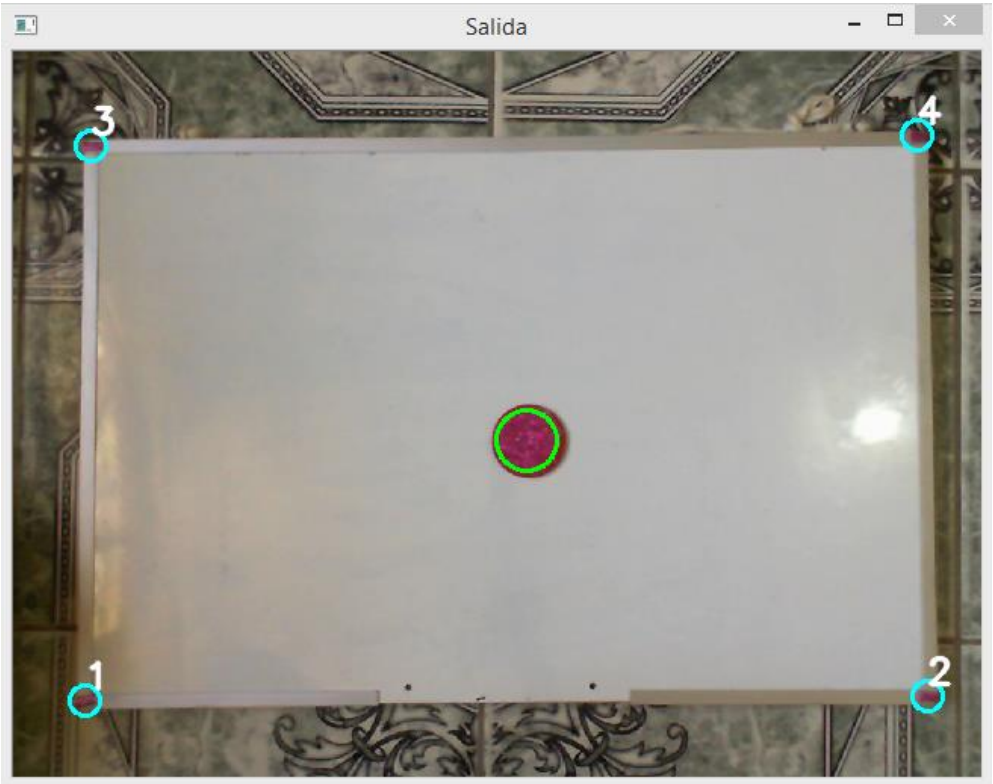

FIGURA 35 LOCALIZACIÓN DE LAS ESQUINAS

3.1.3.1. DELIMITACIÓN DEL AREA

Como ya tenemos los centroides de cada objeto de las esquinas, localizados y segmentados, trazamos líneas para visualmente delimitar el área de trabajo.

Cv2.line(frame,(lz,lf),(ld,lf),(255,0,0),2)

cv2.line(frame,(ld,lf),(ld,ls),(255,0,0),2)

cv2.line(frame,(ld,ls),(lz,ls),(255,0,0),2)

cv2.line(frame,(lz,ls),(lz,lf),(255,0,0),2)

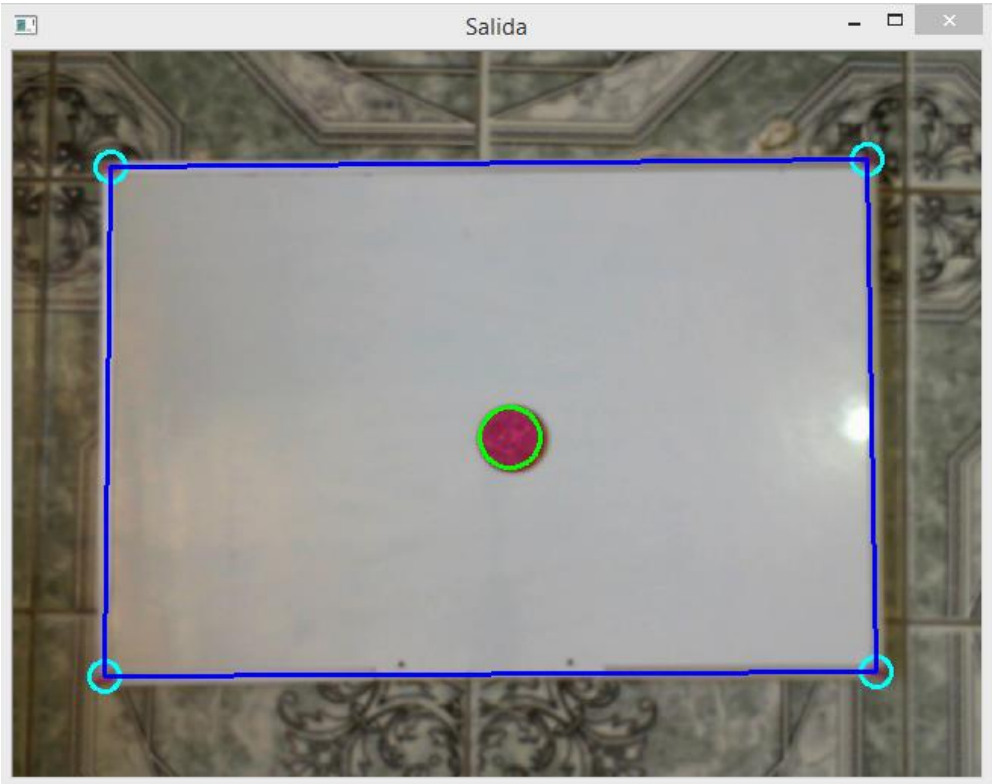

FIGURA 36 DELIMITACIÓN DEL AREA DE TRABAJO

Nos quedamos en memoria de la PC con las coordenadas de estas esquinas, los bordes superior e inferior, nos permitirá saber dónde va a rebotar la esfera cuando siga una trayectoria.

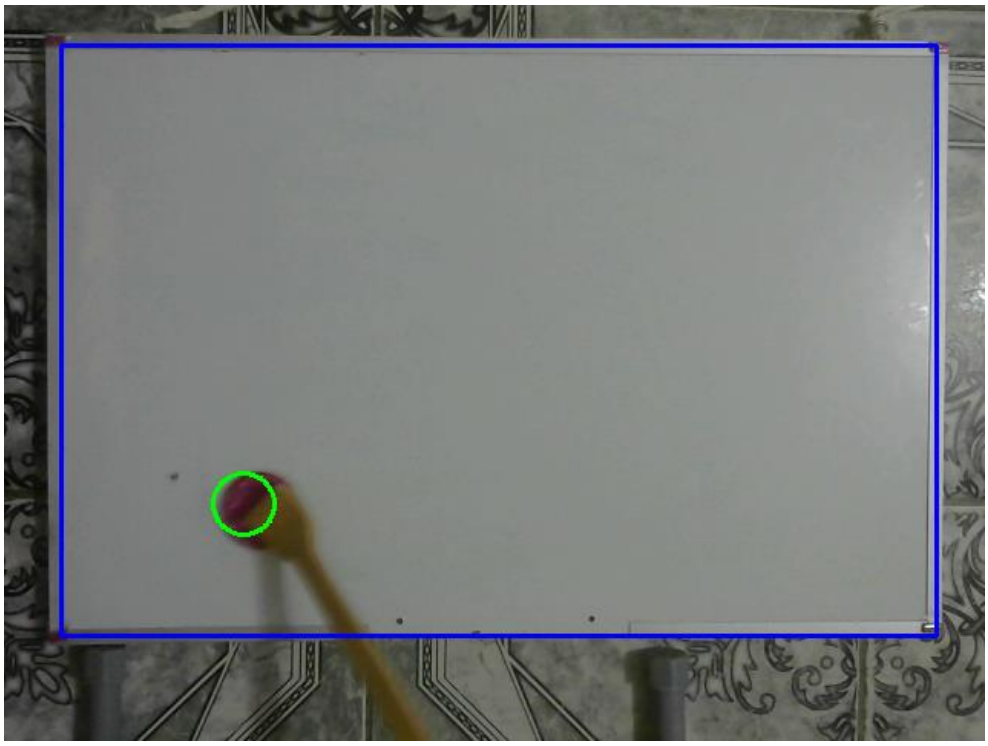

FIGURA 37 DELIMITACIÓN DEL AREA DE TRABAJO SIN DETECCIÓN DE OBJETOS DE ESQUINAS

3.1.3.1. PREDICCIÓN DE POSICIÓN Y TRAYECTORIA DE LA ESFERA

Siguiendo la simulación de Matlab y utilizando las mismas fórmulas, el siguiente paso es deslizar la esfera y obtener secuencia de imágenes para utilizar las posiciones previas de la esfera y poder predecir su posición futura.

```
n = n + 1
sx = sx + esfX
sy = sy + esfY
sxy = sxy + (esfX * esfY)
sxc = sxc + (esfX * esfX)
if(n>2 and n<5):
   m = (sxy - (sx*sy/n)) / ((sxc-(sx*sx/n))+0.0001)
   xp = sx/n
   yp = sy/n
```

```
b = yp - m*xp
for nx in range(esfX,ld):
    ny = int(m*nx + b)
    cv2.circle (frame,(nx,ny),4,(0,255,0), -1)
    if(ny>=lf):
        m = -m;
        b = -b+(2*lf);
    if(ny<=ls):
        m = -m;
        b = -b+(2*ls);
```

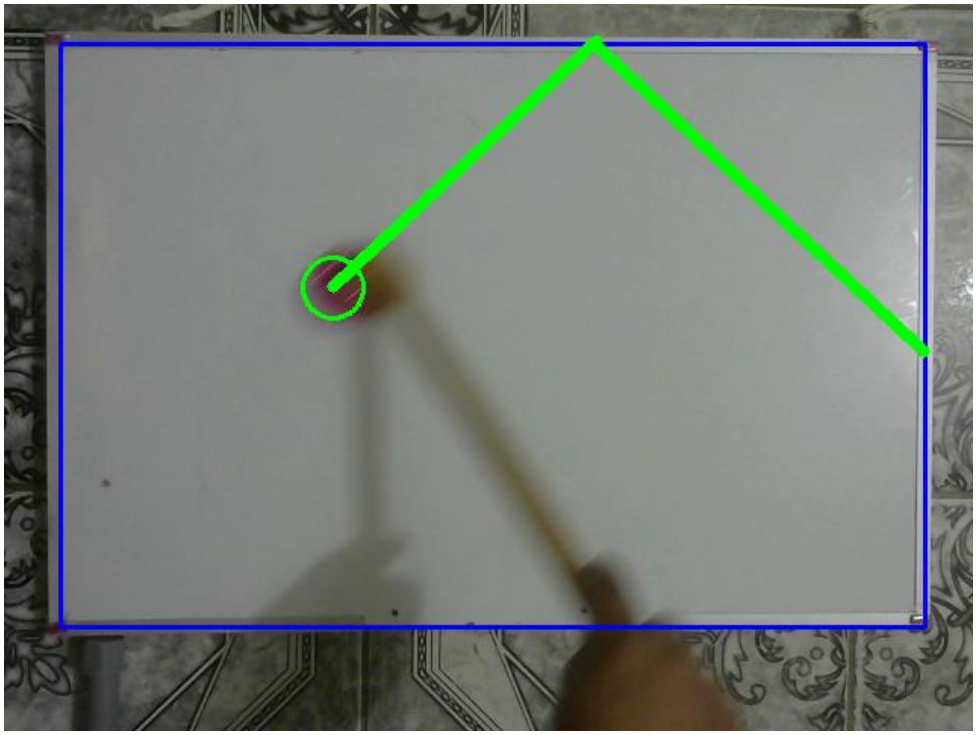

FIGURA 38 PREDICCIÓN DE POSICIÓN Y TRAYECTORIA DE LA ESFERA

IV. RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

El sistema tiene 2 etapas, la primera es la delimitación del área de trabajo y la otra, después de ésta, el seguimiento de la esfera. Se trata al máximo de minimizar el código para que el tiempo de procesamiento del seguimiento de la esfera sea menor. A pesar de esto, el número de imágenes procesadas se reduce a 13 imágenes por segundo, que sí afecta en la detección de los objetos. La esfera tiene que moverse a una velocidad lenta para poder capturar bien la silueta de ésta.

Aun así, se pueden observar en las imágenes siguientes, buenos resultados para predecir la posición de la esfera, se necesita imágenes previas del recorrido de la esfera, éstas se ubican de acuerdo con el procesamiento visto anteriormente

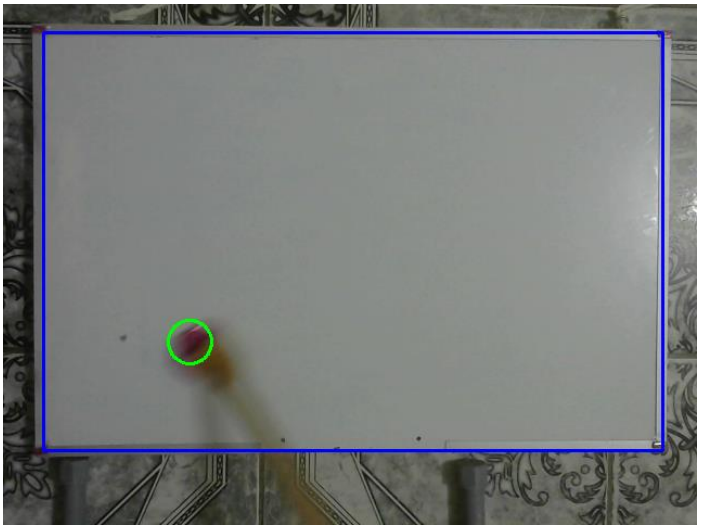

FIGURA 39 IMAGEN PREVIA

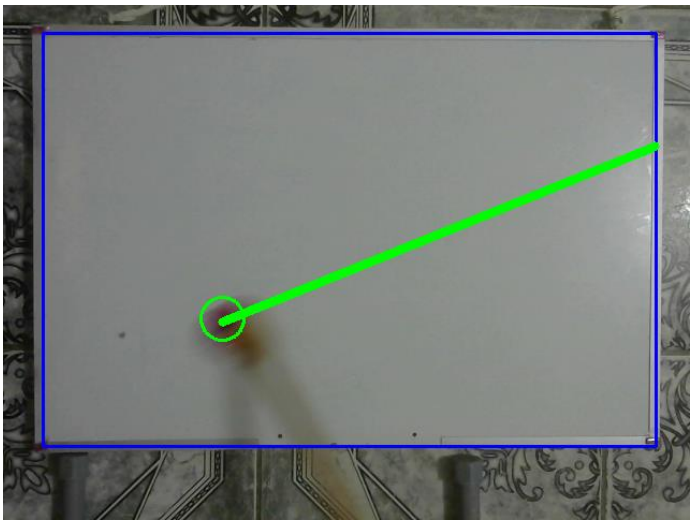

FIGURA 40 ***PREDICCIÓN DE POSICION Y SEGUIMIENTO***

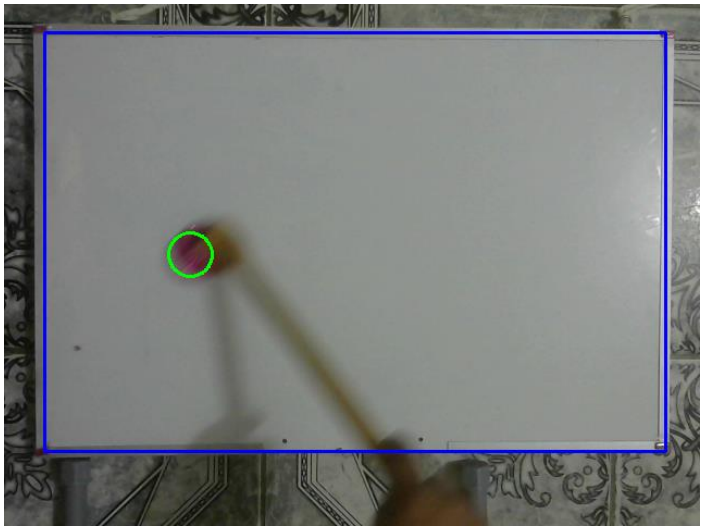

FIGURA 41 ***IMAGEN PREVIA 1***

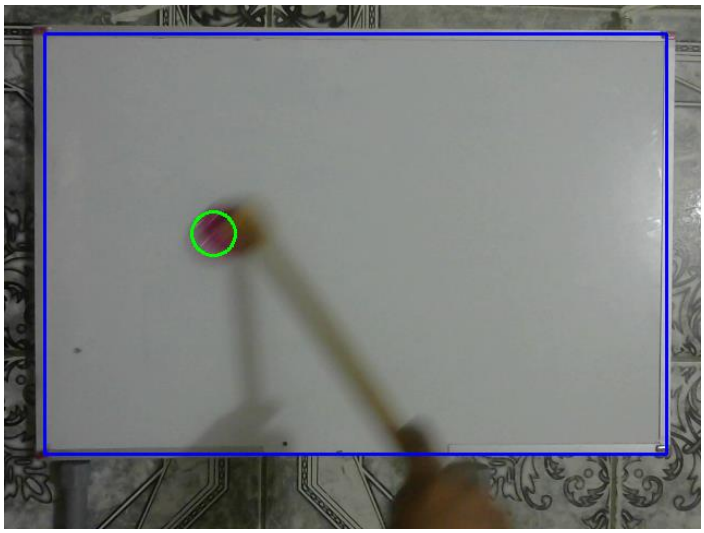

FIGURA 42 ***IMAGEN PREVIA 2***

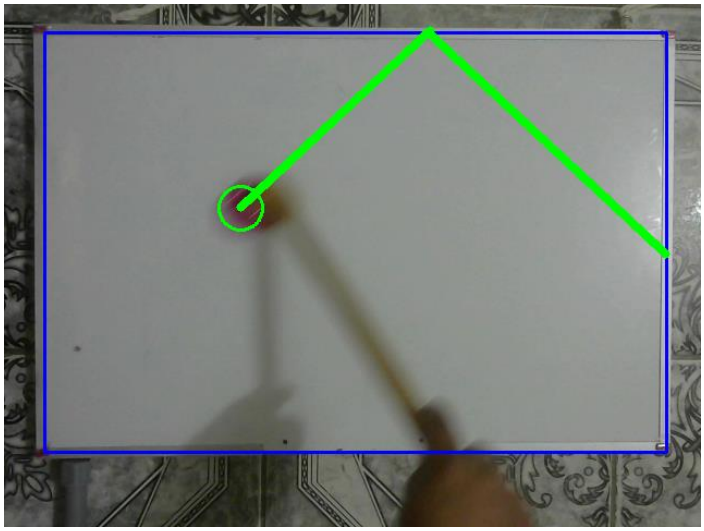

FIGURA 43 ***PREDICCIÓN DE POSICION Y SEGUIMIENTO***

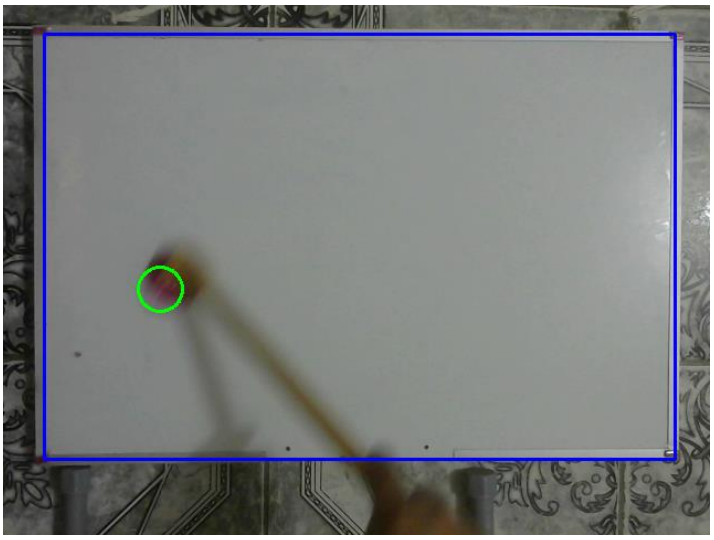

FIGURA 44 *IMAGEN PREVIA 1*

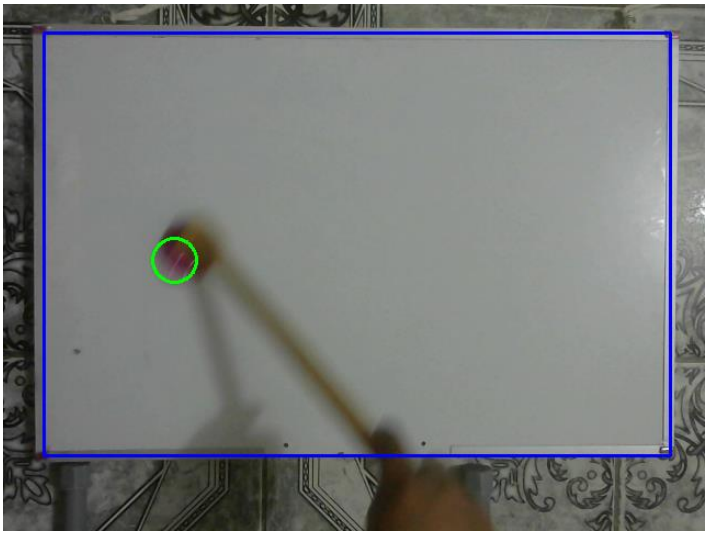

FIGURA 45 *IMAGEN PREVIA 2*

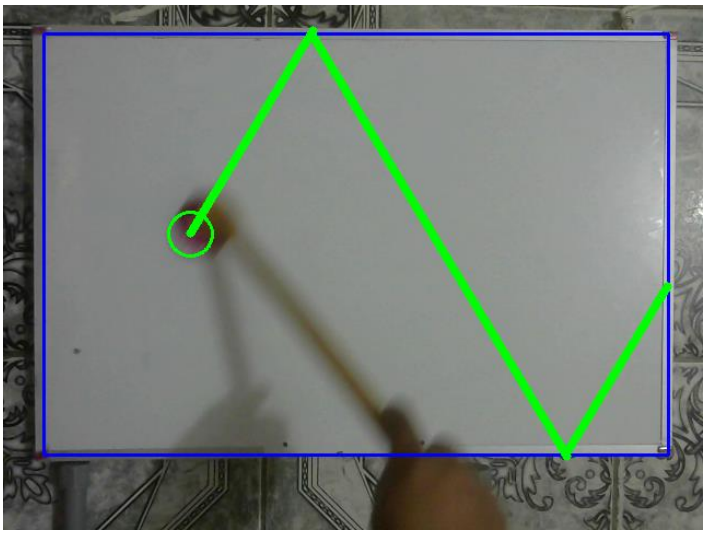

FIGURA 46 ***PREDICCIÓN DE POSICIÓN Y SEGUIMIENTO***

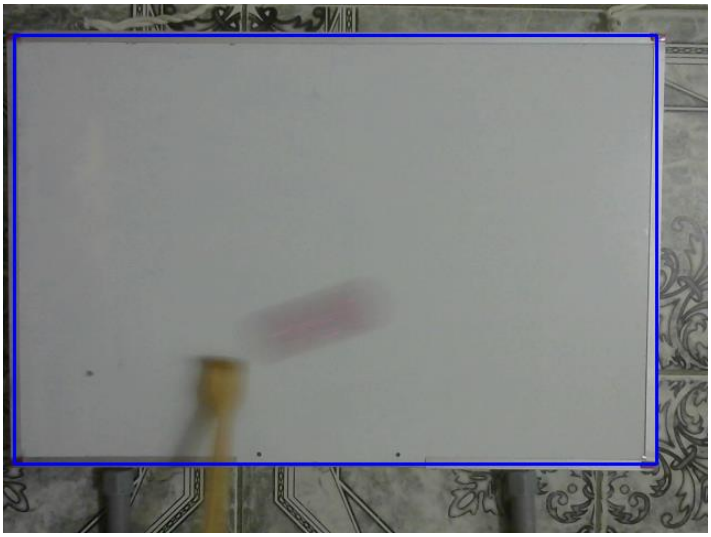

FIGURA 47 ***IMAGEN PREVIA 1***

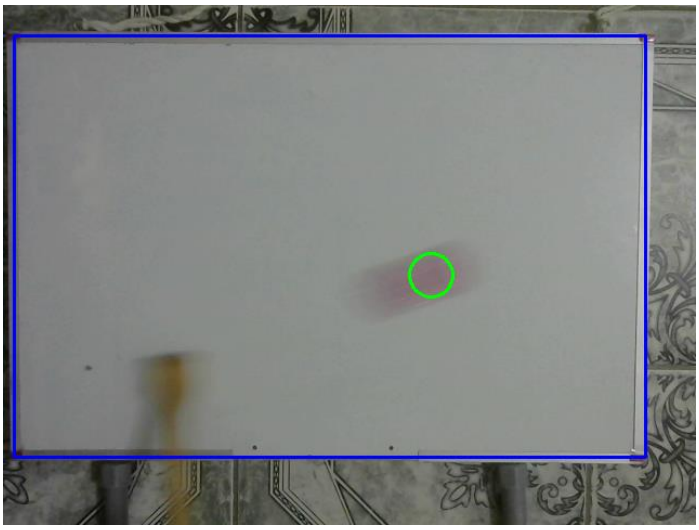

FIGURA 48 *IMAGEN PREVIA 2*

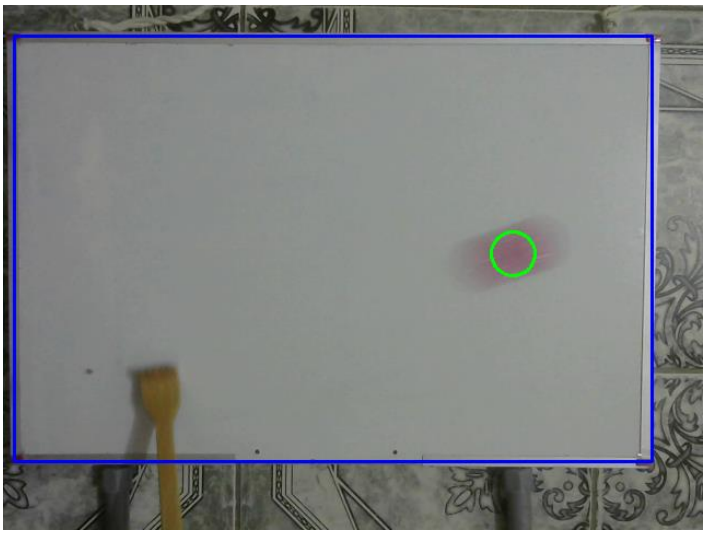

FIGURA 49 *IMAGEN PREVIA 3*

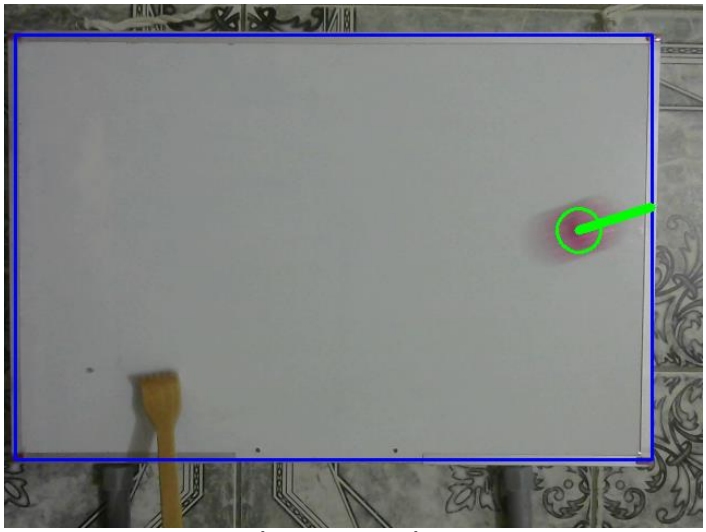

FIGURA 50 ***PREDICCIÓN DE POSICIÓN Y SEGUIMIENTO***

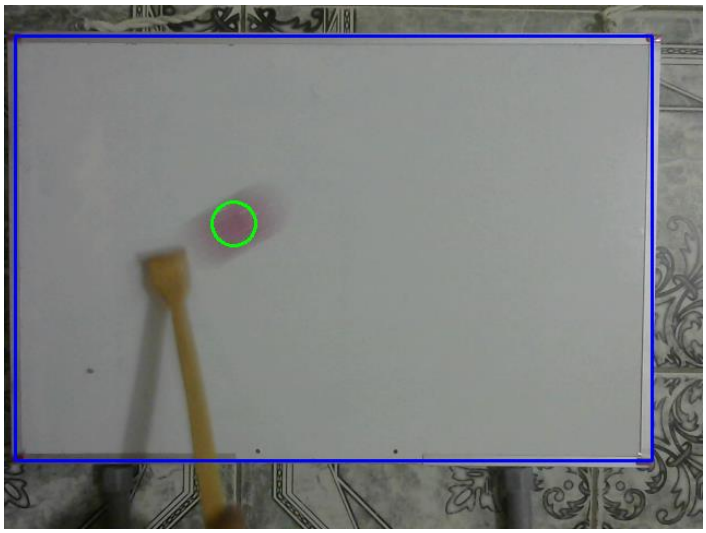

FIGURA 51 ***IMAGEN PREVIA 1***

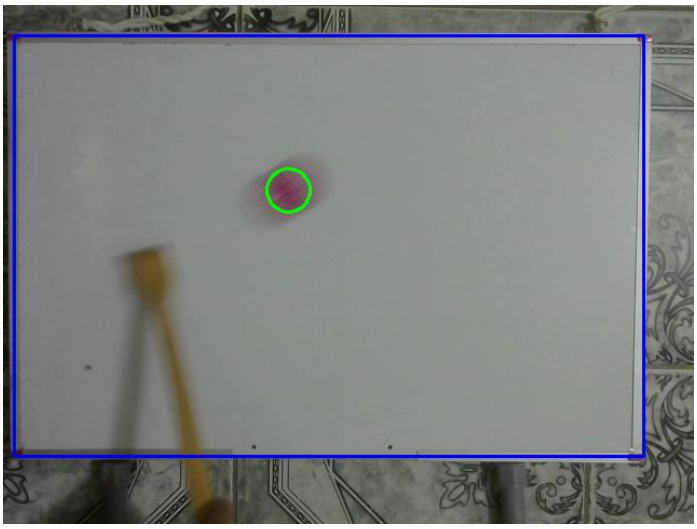

FIGURA 52 *IMAGEN PREVIA 2*

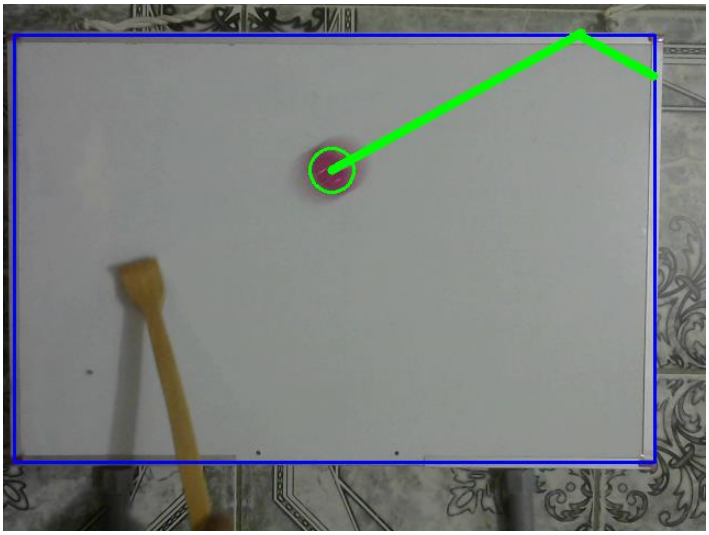

FIGURA 53 *PREDICCIÓN DE POSICIÓN Y SEGUIMIENTO*

4.2. DISCUSIÓN

La reducción de frames por segundo, debido a la captura y procesamiento de las imágenes, se debe al lenguaje de programación utilizado, Python es un lenguaje de programación interpretado, es decir, no se compila, como si lo hace por ejemplo Visual C++.

Otro factor importante a tener en cuenta para obtener más frames por segundo es justamente una buena cámara. Hay cámaras comerciales que capturan a 60 o 120 frames por segundo, en comparación a las convencionales que solo lo hace a 30 cuadros por segundo.

CONCLUSIONES

Se logró diseñar un sistema de predicción de posición y seguimiento de una esfera en tiempo real utilizando visión artificial.

Se han implementado y analizado los algoritmos que resuelven el problema bajo el modelo de un solo dispositivo de captura, cámara webcam y una PC para el procesamiento. Las pruebas se realizaron con datos generados artificialmente simulando el comportamiento de una cámara real, con el software científico Matlab. Esto último simplifica las primeras etapas del sistema, que son las relacionadas con la captura y mejora de los datos.

La implementación de este tipo de sistemas se reduce a la correcta ubicación de una cámara y en algunos casos a una estable iluminación lo que representa una gran ventaja frente a sistemas con sensores convencionales mucho más intrusivos que el enfoque de una cámara.

La segmentación de los objetos se logró con el modelo HSV. Modelo con el que se obtuvieron buenos resultados.

La predicción se logró utilizando el método de mínimos cuadrados. Se obtuvo la pendiente y el intercepto con el eje y. Con estos parámetros se obtuvo la ecuación de la recta y con esta se logró predecir la posición de la esfera.

La Validación del algoritmo desarrollado se visualiza en pantalla de las imágenes procesadas y se compara con el seguimiento de la esfera.

Para desarrollar el sistema se utilizó el lenguaje de programación Python, que demostró ser un sistema amigable y sencillo de programar. Por ser un lenguaje de programación interpretado lo hace ligeramente menos veloz en cuanto a tiempo de procesamiento que Visual C++.

Las librerías de Opencv como herramienta de tratamiento y manejo de imágenes son muy útiles debido a su fácil implementación, su considerable versatilidad la cual permite abrir, crear, visualizar y modificar archivos tanto de imagen y video, el reducido costo computacional de cada una de sus funciones y su importante característica de ser un código tipo open source, lo que significa la ausencia de algún costo asociado y una enorme comunidad de apoyo y soporte técnico.

RECOMENDACIONES

Se deja como recomendación a partir de este trabajo que se estudien diferentes técnicas que permitan estimar otros tipos de movimiento (rectilíneos uniformemente acelerados, circulares, curvilíneos, etc.). Es importante establecer alguna métrica que permita decidir cual modelo de movimiento es conveniente utilizar para realizar la predicción cuando en el escenario analizado existen entidades que se mueven de distintas maneras.

También se deja como recomendación para aumentar la velocidad de procesamiento, utilizar más de una PC y distribuir el procesamiento.

REFERENCIAS BIBLIOGRÁFICAS

1. MERY, Domingo. Visión por Computador. Departamento de ciencia de la Computación Universidad Católica de Chile. Santiago de Chile, 2004
2. Rafael C. Gonzales, Richard E. Woods, Tratamiento Digital de Imágenes. USA, Addison Wesley Iberoamericana, S. A. 1996. 755 pp.
3. Michael Alder, An Introduccion to Pattern Recognition, Mike Alder, 2001.
4. Abraham Kandel, Horst Bunke, Mark Last, Applied graph theory in computer visión and pattern recognition, Springer-Verlag Berlin Heidelberg, 2007.
5. Bernd Jahne, Digital image processing, Springer-Verlag Berlin Heidelberg, 2002.
6. Gonzalo Pajares Martinsanz, Jesús Manuel de la Cruz García, José Manuel Molina Pascual, Juan Cuadrado Pardo, Alejandro López Correa, "Imágenes digitales - Procesamiento practico con java, Alfa Omega Grupo Editor S.A. 2004
7. Computer Vision with the OpenCV library; Gary Bradsky y Adrian Kaebler.
8. Manual original proporcionado por intel: Intel® Open Source Computer Vision Library.
9. Learning OpenCV: Computer Vision with the OpenCV Library, Bradski, G.,
10. BRADSKY, Gary, KAEHLER, Adrián. Learning OpenCv.

ANEXOS

ANEXO 1CÓDIGO DEL PROGRAMA

```
import numpy as np
import cv2

# Creamos una variable de camara y asigamos la primera camara disponible con "0"
cap = cv2.VideoCapture(1)
# Asignamos las variables del rango de color que seguiremos
Hmin = 132
Hmax = 256
Smin = 67
Smax = 256
Vmin = 40
Vmax = 256
pic = 0
ancho = 640
alto = 480
direc = 0
esfXa = 0
n = 0
sx = 0
sy = 0
sxy = 0
sxc = 0
lz = 0
ld = 0
ls = 0
lf = 0
def_area = 0
num = 0
vX = [0,0,0,0,0,0]
vY = [0,0,0,0,0,0]
```

```
esfX = 0
esfY = 0
# Iniciamos el bucle de captura, en el que leemos cada frame de la captura
while(True):
   ret, frame = cap.read()
   hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV) #Convertimos imagen a HSV

   # Aqui mostramos la imagen en blanco o negro segun el rango de colores.
   bn_img = cv2.inRange(hsv, np.array((Hmin,Smin,Vmin)), np.array((Hmax,Vmax,Smax)))

   # Limpiamos la imagen de imperfecciones con los filtros erode y dilate
   bn_img = cv2.erode(bn_img,cv2.getStructuringElement(cv2.MORPH_RECT,(3,3)),iterations = 1)
   bn_img = cv2.dilate(bn_img,cv2.getStructuringElement(cv2.MORPH_RECT,(5,5)),iterations = 1)
   cnts = cv2.findContours(bn_img.copy(), cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
   # Localizamos la posicion del objeto
   cnt = cnts[1]

   for c in cnt:
      area = cv2.contourArea(c)
      M = cv2.moments(c)
      cx = int(M['m10']/M['m00'])
      cy = int(M['m01']/M['m00'])
      if area>20 and area<150:
         if(def_area == 0):
            cv2.circle(frame,(cx,cy),10,(255,255,0), 2)
            vX[num] = cx
            vY[num] = cy
            num = num + 1
            if(num > 3):
```

```
            num = 0
    if area>200:
        esfX = cx
        esfY = cy
    # Mostramos un circulo verde en la posicion en la que se encuentra el objeto
        cv2.circle (frame,(cx,cy),20,(0,255,0), 2)

if cv2.waitKey(1) & 0xFF == ord('p'):
    def_area = 1

if (def_area == 0):
    if(vY[0]<vY[1]+100 and vY[0]>vY[1]-100):
        cv2.line(frame,(vX[0],vY[0]),(vX[1],vY[1]),(255,0,0),2)
        lf = vY[0]

    if(vY[2]<vY[3]+100 and vY[2]>vY[3]-100):
        cv2.line(frame,(vX[2],vY[2]),(vX[3],vY[3]),(255,0,0),2)
        ls = vY[2]

    if(vX[0]<=vX[3]+100 and vX[0]>=vX[3]-100):
        cv2.line(frame,(vX[0],vY[0]),(vX[3],vY[3]),(255,0,0),2)
        if(vX[0]<ancho/2):
            lz = vX[0]
        else:
            lz = vX[3]

    elif(vX[1]<=vX[3]+100 and vX[1]>=vX[3]-100):
        cv2.line(frame,(vX[1],vY[1]),(vX[3],vY[3]),(255,0,0),2)
        if(vX[1]>ancho/2):
            lz = vX[1]
        else:
            lz = vX[3]
```

```
    if(vX[0]<=vX[2]+100 and vX[0]>=vX[2]-100):
        cv2.line(frame,(vX[0],vY[0]),(vX[2],vY[2]),(255,0,0),2)
        if(vX[0]<ancho/2):
            ld = vX[0]
        else:
            ld = vX[2]

    elif(vX[1]<=vX[2]+100 and vX[1]>=vX[2]-100):
        cv2.line(frame,(vX[1],vY[1]),(vX[2],vY[2]),(255,0,0),2)
        if(vX[1]>ancho/2):
            ld = vX[1]
        else:
            ld = vX[2]

if (def_area == 1):
    cv2.line(frame,(lz,lf),(ld,lf),(255,0,0),2)
    cv2.line(frame,(ld,lf),(ld,ls),(255,0,0),2)
    cv2.line(frame,(ld,ls),(lz,ls),(255,0,0),2)
    cv2.line(frame,(lz,ls),(lz,lf),(255,0,0),2)

direc = esfX - esfXa

if(direc>10):
    n = n + 1
    sx = sx + esfX
    sy = sy + esfY
    sxy = sxy + (esfX * esfY)
    sxc = sxc + (esfX * esfX)
    if(n>2 and n<5):
        m = (sxy - (sx*sy/n)) / ((sxc-(sx*sx/n))+0.0001)
        xp = sx/n
        yp = sy/n
        b = yp - m*xp
```

```
            for nx in range(esfX,ld):
                ny = int(m*nx + b)
                cv2.circle (frame,(nx,ny),4,(0,255,0), -1)
                if(ny>=lf):
                    m = -m;
                    b = -b+(2*lf);
                if(ny<=ls):
                    m = -m;
                    b = -b+(2*ls);
            n = 0
            sx = 0
            sy = 0
            sxy = 0
            sxc = 0

    if(def_area == 0):
        cv2.imshow('inRange', bn_img)
    cv2.imshow('Salida', frame)
    esfXa = esfX

    if cv2.waitKey(1) & 0xFF == ord('q'): # Indicamos que al pulsar "q" el programa se
cierre
        break

cap.release()
cv2.destroyAllWindows()
```

```
%%
% Metodo de minimos cuadrados para determinar
% la ecuacion de la recta
x = [8 2 11 6 5 4 12 9 6 1];
y = [3 10 3 6 8 12 1 4 9 14];
plot(x,y,'xr')
xlabel('Valores de X');
ylabel('Valores de Y');
grid
axis([-2 14 -2 16])
%%
% Metodo de minimos cuadrados para determinar
% la ecuacion de la recta
x = [8 2 11 6 5 4 12 9 6 1];
y = [3 10 3 6 8 12 1 4 9 14];
plot(x,y,'xr')
grid
axis([-2 14 -2 16])
hold on
% Hallamos la pendiente (m):
n = length(x)
sx = sum(x)
sy = sum(y)
sxy = sum(x.*y)
sxc = sum(x.^2)
m = (sxy - (sx*sy/n)) / (sxc-(sx^2/n))
% para calcular la intercepción en y (b):
xp = sx/n
yp = sy/n
b = yp - m*xp
```

```
%%
% Metodo de minimos cuadrados para determinar
% la ecuacion de la recta
x = [8 2 11 6 5 4 12 9 6 1];
y = [3 10 3 6 8 12 1 4 9 14];
plot(x,y,'xr')
xlabel('Valores de X');
ylabel('Valores de Y');
grid
axis([-2 14 -2 16])
hold on
% Hallamos la pendiente (m):
n = length(x)
sx = sum(x)
sy = sum(y)
sxy = sum(x.*y)
sxc = sum(x.^2)
m = (sxy - (sx*sy/n)) / (sxc-(sx^2/n))
% para calcular la intercepción en y (b):
xp = sx/n
yp = sy/n
b = yp - m*xp
% Ecuacion de la recta es:
nx = 0:12;
ny = m*nx + b
plot(nx,ny,'b')
hold off

%%
lz=0;
ld=50;
ls=30;
lf=0;
```

```
line([lz ld ld ld ld lz lz lz],[lf lf lf ls ls ls ls lf])
grid
hold on
% Ecuacion de la recta es:
m=-1.00; b=14
nx = 0:13;
ny = m*nx + b
plot(nx,ny,'r')

nx = 13:40;
ny = -m*nx + b-26
plot(nx,ny,'r')
hold off
%%
lz=0;
ld=50;
ls=30;
lf=0;
line([lz ld ld ld ld lz lz lz],[lf lf lf ls ls ls ls lf])
axis([lz-5 ld+5 lf-5 ls+5])
xlabel('Valores de X');
ylabel('Valores de Y');
grid
hold on
% Ecuacion de la recta es:
m=-1.20; b=15;
for nx=lz:0.5:ld
    ny = m*nx + b;
    plot(nx,ny,'xr')
    if(ny<=lf)
        m = -m;
        b = -b;
    end
```

```
    if(ny>=ls)
        m = -m;
        b = -b+(2*ls);
    end
end
hold off
```